

**2015 specification**  
for the 2019 exam



**PYTHON2**

# **PAPER 1 EXAM RESOURCE PACK 2019**

## **AQA Text Adventures**

for A Level AQA Computer Science

**CR7/  
8872**

**POD  
8872**

**zigzageducation.co.uk**

Publish your  
own work...  
Write to a brief...  
**Register at**  
**publishmenow.co.uk**

# Contents

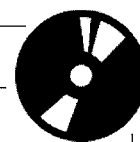
|                                                      |            |
|------------------------------------------------------|------------|
| <b>Thank You for Choosing ZigZag Education .....</b> | <b>ii</b>  |
| <b>Teacher Feedback Opportunity .....</b>            | <b>iii</b> |
| <b>Terms and Conditions of Use .....</b>             | <b>iv</b>  |
| <b>Teacher's Introduction .....</b>                  | <b>v</b>   |
| <b>Printouts of CD resources (for reference)</b>     |            |

- Commentary (20 pages)
- Structure Diagram Activity 1 (1 page)
- Structure Diagram Activity 2 (1 page)
- Structure Diagram Activity 3 (1 page)
- Written Questions: Non-write-on version (1 page)
- Written Questions: Write-on version (4 pages)
- Programming Tasks (9 pages)
- Additional Tasks (Extension) (1 page)
- Structure Diagram Activity 1: Solution (1 page)
- Structure Diagram Activity 2: Solution (1 page)
- Structure Diagram Activity 3: Solution (1 page)
- Written Questions: Mark Scheme (2 pages)
- Programming Tasks: Mark Scheme (18 pages)
- Electronic Answer Document (3 pages)

# Teacher's Introduction

This resource pack is designed to help you support your students taking the **A Level Computer Science Paper 1** examination. It is based on the 'AQA Text Adventures' preliminary material (Python2) – for examination June 2019.

On the CD, you will find the following files.



|                   |                                                                                                      |
|-------------------|------------------------------------------------------------------------------------------------------|
| AQATextAdventures | for student use – this folder contains all of the content, accessible via a HTML interface           |
| editable          | for teacher use – this folder contains ALL of the documents in editable (docx/pptx) formats          |
| Passwords.txt     | for teacher use – this file contains all of the passwords for the protected PDFs (also listed below) |

\* PRINTED COPIES OF ALL THE MATERIALS IN THIS DIGITAL RESOURCE PACK ARE INCLUDED FOR REFERENCE.

**Installation:** Copy the entire `AQATextAdventures` folder onto a network location that is accessible for students, and provide them with a shortcut to the `index.html` file. All content can be accessed from this page.

**Passwords:** All of the PDFs in the 'Answers & Solutions' HTML page (`answers.html`) are password-protected, so that students can only access them with your permission. Each password is a four-digit code, as follows:

|                         |      |
|-------------------------|------|
| Commentary.pdf          | 1158 |
| Diagram1Complete.pdf    | 4773 |
| Diagram2Complete.pdf    | 5382 |
| Diagram3Complete.pdf    | 3091 |
| QuestionsMarkScheme.pdf | 7642 |
| TaskMarkScheme.pdf      | 2966 |

Should you wish to give students access to ALL protected-PDFs, the master password for all files is:  
`zz2ghc4`

The resource pack consists of the following:

① **Pre-release Commentary**, consisting of two parts:

- A general walkthrough of the skeleton program; a written description, flowchart and an animated PowerPoint giving a visual demonstration of the game. It is non-technical in the sense that it doesn't reference or explain any actual code elements – only how the program works when it is run.
- A detailed, technical overview of the skeleton program, describing how all Python subroutines, classes and variables work, including the relationship between them (also shown visually as a structure diagram).

**Note:** although this section is intended to give extra support to teachers and students, it should in no way be seen as a substitute to a student exploring the code for themselves. For this reason, this content has been placed on the 'Answers & Solutions' HTML page as a password-protected file, to allow you to control if/when students access it.

② **Structure Diagram Activities**

Three partially complete structure diagram activities for students to complete while getting to grips with the skeleton program. Any missing identifiers, data types, return values, directional arrows, etc. must be added to the diagram. Solutions are provided on the *Answers & Solutions* page as a protected PDF.

③ **Written Questions**

Theory questions testing students' understanding of the skeleton program, like Section C in the exam. These questions require access to the program, but no modifications need to be made to the program. Write-on (with answer lines) and non-write-on version are available format. Solutions are provided on the *Answers & Solutions* page as a protected PDF.

④ **Programming Tasks**

Fifteen modification exercises put students' programming skills to the test, like Section D in the exam. Solutions are provided on the *Answers & Solutions* page as a protected PDF. Note that these are example solutions and you must use your discretion to award marks accordingly where there are valid alternative solutions.

### Free Updates

Register your email address to receive any future free minor updates made to this resource or other Computing resources your school has purchased, and details of any promotions for your subject.

*\* resulting from minor specification changes, suggestions from teachers and peer reviews, or occasional errors reported by customers*

[zzed.uk/freeupdates](http://zzed.uk/freeupdates)

An **Electronic Answer Document (EAD)** is provided should you wish students to use it for ③ and/or ④ above.

This resource is intended to supplement your teaching only. **Please read full disclaimer (p. iv) before using it.**

# AQA Text Adventure

## Introduction

AQA Text Adventures is a text adventure game, where the player enters text commands, moves between locations, use various items and interact with different characters. The program displays the result of their actions and the current state of the game.

When the program starts, there are no items, characters or places in the game. The game file (which will have the .gme file extension) they would like to use. If the program cannot find the file, a message will be displayed to the player telling them that the game could not be found and the game will terminate. If the file is found, then all of the items, characters and places in that file are loaded and the game runs as below.

1. The game checks whether the player has moved to a new place. If they have, then the game displays the description of that place and the items in that place which the player can take.
2. The game asks the player to input an action.
3. The game identifies the command by splitting up the input into the command and the argument (e.g. if the player enters "get torch", the game will split this into the command "get" and the argument "torch").
4. If the command is recognised, the game will perform the corresponding action and display the result.

The recognised commands are as follows.

|          |                                                                                            |
|----------|--------------------------------------------------------------------------------------------|
| get      | Gets the specified item to the player inventory                                            |
| use      | Uses an item in the current place or the player's inventory                                |
| go       | Moves the player from their current place to the place in the game                         |
| read     | Displays the text of a readable item in the current place or the player's inventory        |
| examine  | Provides a description of the specified item, place or character in the player's inventory |
| open     | Opens the specified item in the current place                                              |
| close    | Closes the specified item in the current place                                             |
| move     | Moves the specified item in the current place                                              |
| say      | Prints the specified input to the screen                                                   |
| playdice | Plays a dice game with the specified character in the current place                        |
| quit     | Ends the game                                                                              |

5. If the game does not recognise the command, a message is displayed to say that the command is not recognised.

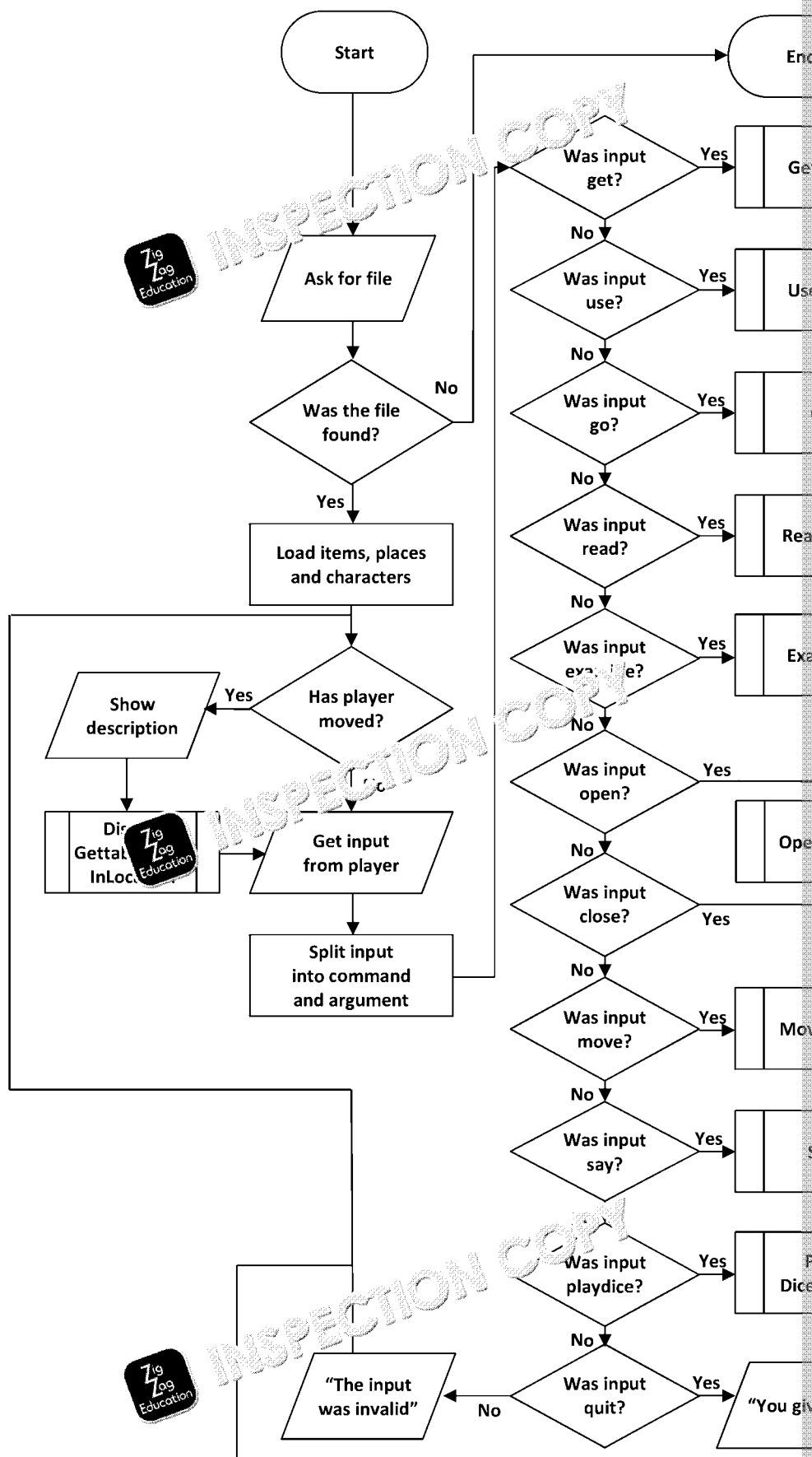
This loop continues until the player quits the game, or gets a particular item that is a flag in the provided .gme files). If the player quits the game, a message is given up, and if the player gets the flag (or other winning item), a message is displayed.

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



# Program Flow Chart



INSPECTION COPY

COPYRIGHT  
PROTECTED

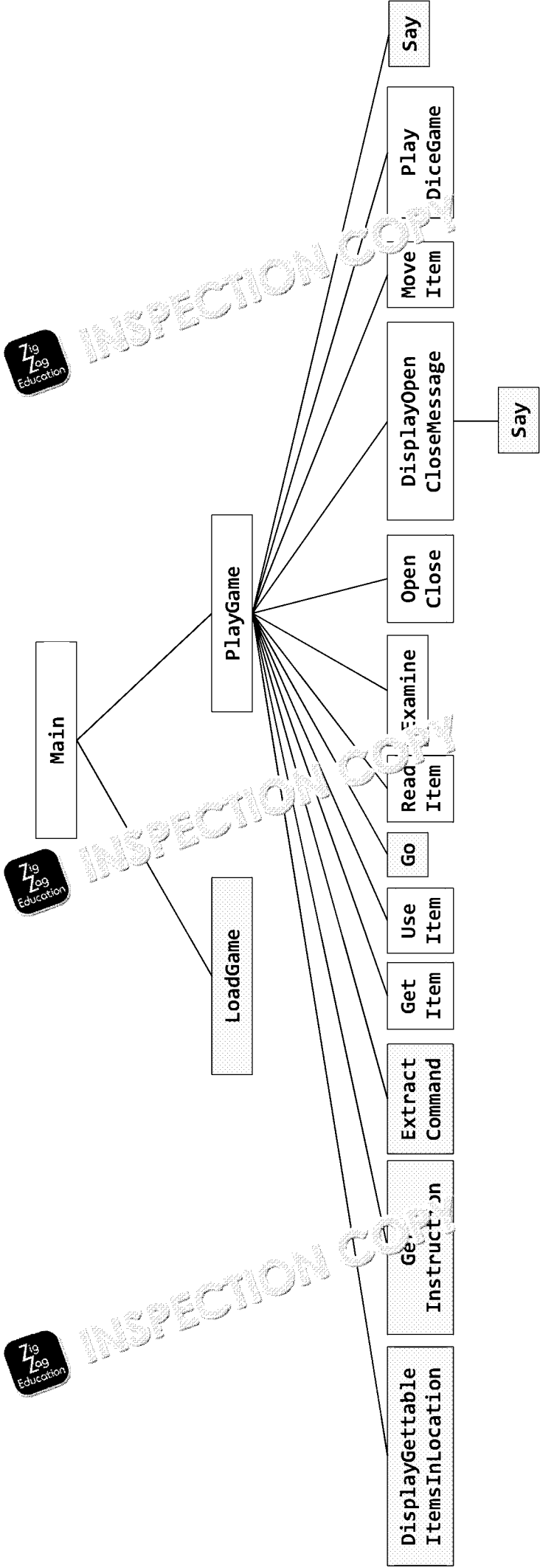


Program Structure Diagrams

Key

Subroutine that calls other subroutine(s)

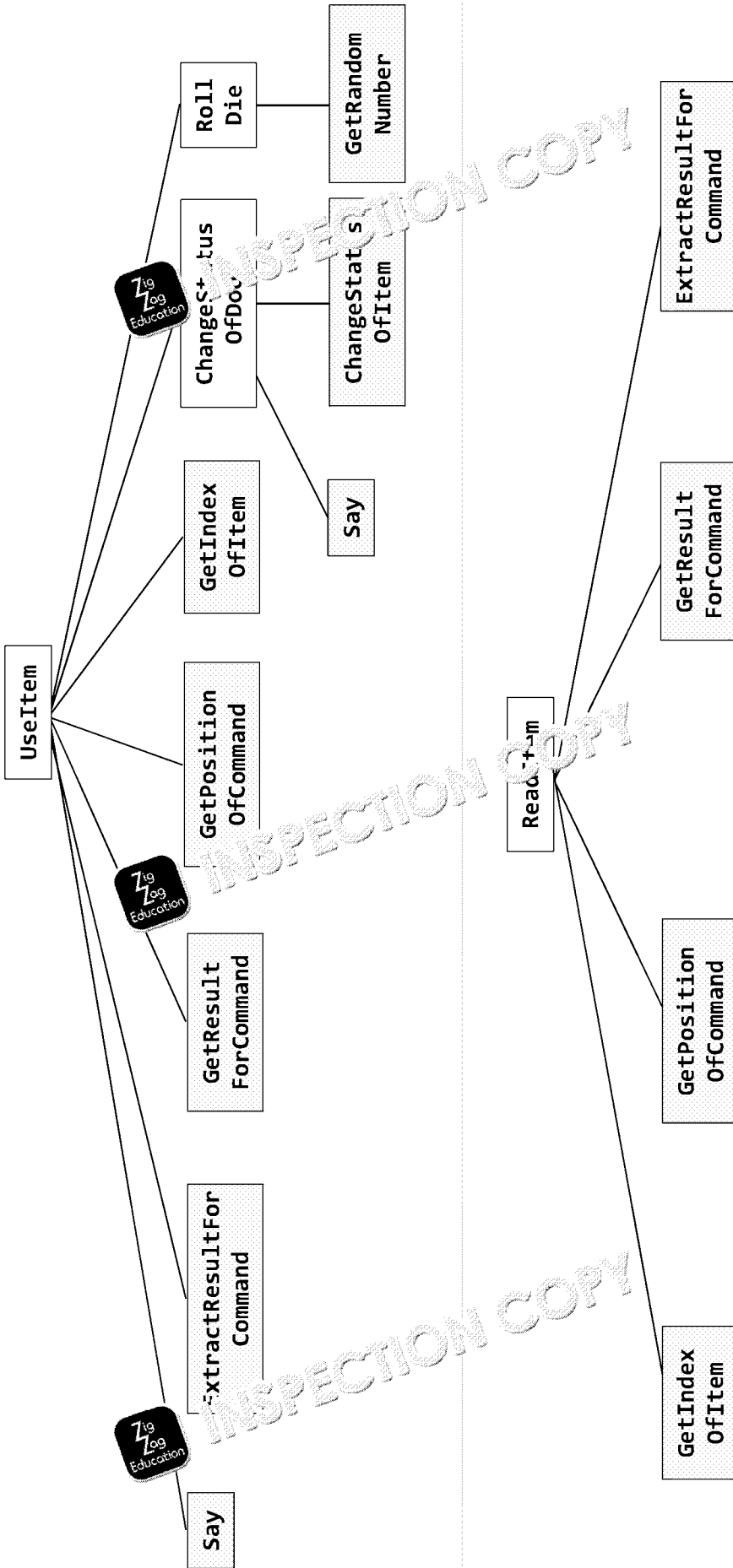
Subroutine that doesn't call other subroutines



COPYRIGHT  
PROTECTED



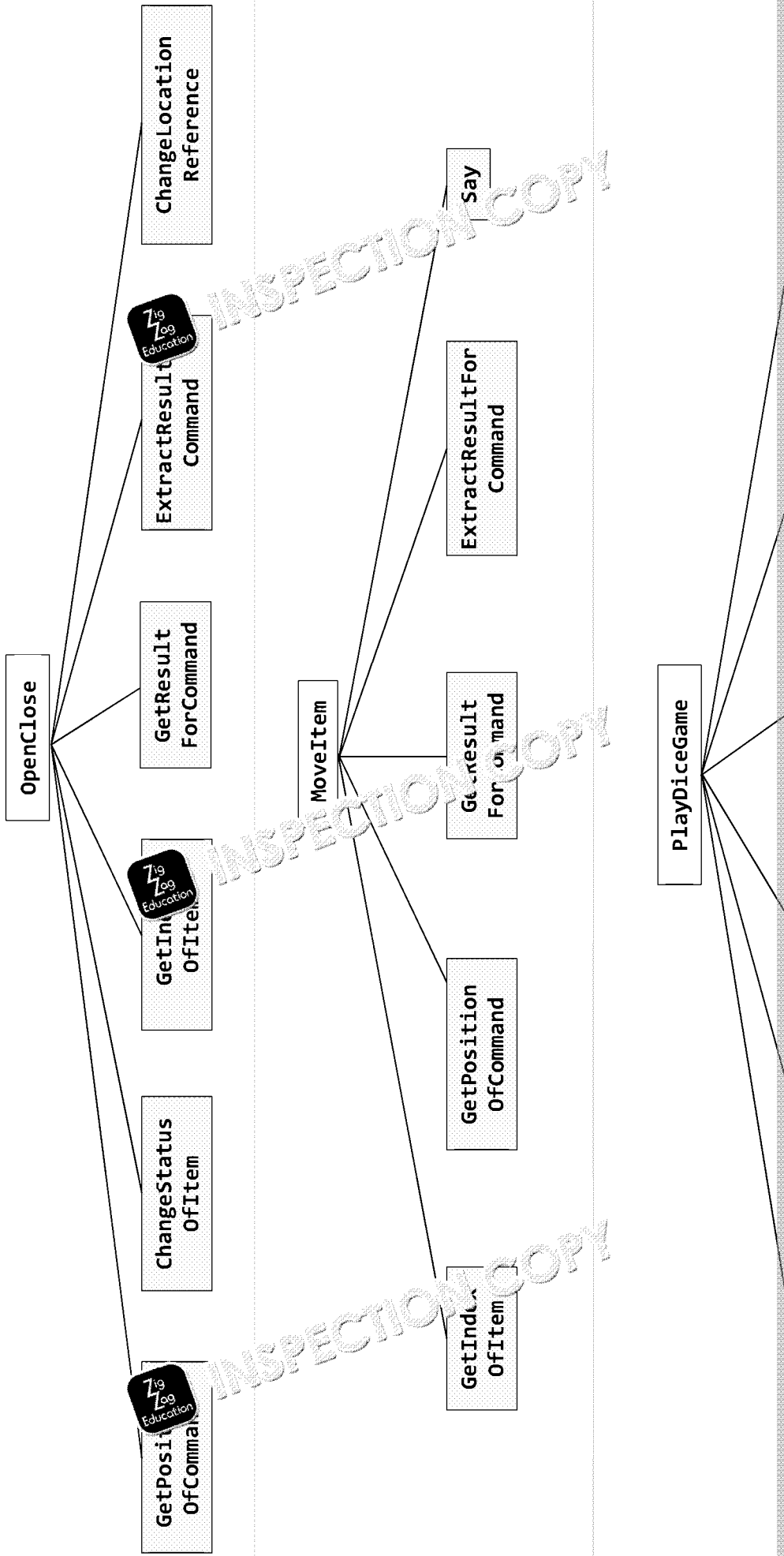
INSPECTION COPY



COPYRIGHT  
PROTECTED



INSPECTION COPY



COPYRIGHT  
PROTECTED



INSPECTION COPY

Program Subroutines

The program's functions ⑥ and procedures ⑦ are described below.

| Subroutine       | Data                                                                                            | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------------|-------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| GetInstruction ⑥ | Parameters:<br>Returns: Instruction (string)<br>Called From: PlayGame<br>Calls: -               |   <ol style="list-style-type: none"><li>1. Prints a line separator</li><li>2. Creates the variable Instruction, which is set to the value of the user's input converted to lowercase</li><li>3. Returns Instruction</li></ol>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| ExtractCommand ⑥ | Parameters:<br>Returns: Command (string)<br>Called From: PlayGame<br>Calls: -                   |   <ol style="list-style-type: none"><li>1. If Instruction doesn't contain a space, then returns Instruction as both the Command and Instruction variables</li><li>2. Otherwise the characters in Instruction are added to the Command variable until a space is found in Instruction</li><li>3. Instruction is set to all of the characters in the Instruction that come after the first space</li><li>4. Returns Command and Instruction</li></ol>                                                                                                                                                                                                                                                              |
| Go ⑦             | Parameters: You (Character)<br>Returns: Direction (string)<br>Called From: PlayGame<br>Calls: - |   <ol style="list-style-type: none"><li>1. Creates the variable Moved and sets it to True</li><li>2. Direction is checked to see if it is a valid direction: "north", "east", "south", "west", "up" or "down"</li><li>3. If a valid direction is given and CurrentPlace is adjacent to another place in that direction, the location of You is set to point to that other location</li><li>4. If a valid direction is given and CurrentPlace is not adjacent to another place in that direction, the value of Moved is set to False</li><li>5. If an invalid direction is given, the value of Moved is set to False</li><li>6. If Moved is False, displays the message "You are not able to go in that</li></ol> |

COPYRIGHT  
PROTECTED



INSPECTION COPY

| Subroutine                                                                                                              | Data                                                                                                                                                                                                                                                 | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DisplayContentsOfContainerItem @<br> | Parameters: Items ([Item])<br>ContainerID (int)<br>Returns: -<br>Called From: Examine<br>Calls: -                                                                                                                                                    | <ol style="list-style-type: none"> <li>1. Displays the message "It contains: " (end = "" stops the next message from being displayed on a new line)</li> <li>2. Creates the variable ContainsItem and sets it to <i>False</i></li> <li>3. Loops through all of the items in the container, and checks whether each item is in the location ContainsItemID</li> <li>4. If the item is in the location ContainsItemID and ContainsItem is <i>True</i>, prints "", sets ContainsItem to <i>True</i> and prints the name of the item</li> <li>5. If the item is in the location ContainsItemID and ContainsItem is <i>False</i>, sets ContainsItem to <i>True</i> and prints the name of the item</li> <li>6. If ContainsItem is <i>True</i>, prints ""</li> <li>7. If ContainsItem is <i>False</i>, prints "nothing"</li> </ol> |
| Examine @<br>                        | Parameters: Items ([Item])<br>Characters ([Character])<br>ItemToExamine (string)<br>CurrentLocation (int)<br>Returns: -<br>Called From: PlayGame<br>Calls: DisplayInventory<br>GetIndexofItem<br>DisplayDoorStatus<br>DisplayContentsOfContainerItem | <ol style="list-style-type: none"> <li>1. Creates the variable Count and sets it to 0</li> <li>2. If ItemToExamine is "inventory", displays inventory</li> <li>3. Otherwise, gets the index of the item</li> <li>4. If the index of the item can be found and the item is in the player's inventory or current location, displays the description of the item</li> <li>5. Once the item description has been displayed, if the item contains "door", the subroutine displays the status of the door<br/>             otherwise, if the status of the item contains "container", the subroutine returns to PlayGame after displaying the contents of the container</li> <li>6. Loops through each character in Characters and if the character's</li> </ol>                                                                   |

COPYRIGHT  
PROTECTED



INSPECTION COPY

| Subroutine                                                                                                    | Data                                                                                                                                                                                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| GetPositionOfCommand ©<br> | Parameters: CommandList (string)<br>Command (string)<br>Returns: Position (int)<br>Called From: GetItem<br>UseItem<br>ReadItem<br>MoveItem<br>OpenClose<br>PlayDiceGame<br>Calls: -      | <ol style="list-style-type: none"> <li>Creates the variable Position and sets it to 0</li> <li>Loops through the characters in CommandList, using each character as the starting position for a substring of CommandList that is the length of Command</li> <li>If the substring is equal to Command, Position is returned</li> <li>If the substring begins with " ", the value of Position is increased by 1</li> <li>Once all substrings have been checked, if Command was not found, Position is returned</li> </ol> |
| GetResultForCommand ©<br>  | Parameters: Results (string)<br>Position (int)<br>Returns: ResultForCommand (string)<br>Called From: GetItem<br>UseItem<br>ReadItem<br>MoveItem<br>OpenClose<br>PlayDiceGame<br>Calls: - | <ol style="list-style-type: none"> <li>Creates the variable ResultForCommand and sets it to ""</li> <li>Loops through each character in Results until as many ","s have been read as the value of Position or the end of the string has been reached</li> <li>Any remaining characters, up until the next " " or the end of the string, are loaded into ResultForCommand</li> <li>ResultForCommand is returned</li> </ol>                                                                                               |
| Say ©<br>                  | Parameters: Speech (string)<br>Returns: -<br>Called From: PlayGame<br>GetItem<br>UseItem<br>MoveItem                                                                                     | <ol style="list-style-type: none"> <li>Prints empty line</li> <li>Prints Speech</li> <li>Prints empty line</li> </ol>                                                                                                                                                                                                                                                                                                                                                                                                   |

**COPYRIGHT  
PROTECTED**



INSPECTION COPY

| Subroutine               | Data                                                                                                                                                                                                                                                                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ExtractResultForCommand® | <p><b>Parameters:</b> SubCommand(string)<br/>SubCommandParameter(string)<br/>ResultForCommand(string)</p> <p><b>Returns:</b> SubCommand(string)<br/>SubCommandParameter</p> <p><b>Called From:</b> GetItem<br/>UseItem<br/>ReadItem<br/>MoveItem<br/>OpenClose<br/>_</p> | <p>1. Loads each character from ResultForCommand into SubCommand in sequence until a “” is read or the end of the string is reached</p> <p>2. Any remaining characters are loaded into SubCommandParameter of the string, are loaded into SubCommandParameter</p> <p>3. Returns SubCommand and SubCommandParameter</p>                                                                                                                                                                                                                                                                                                                                                                                 |
| ChangeLocationReference® | <p><b>Parameters:</b> Direction(string)<br/>NewLocationReference(int)<br/>Places([Place])<br/>IndexOfCurrentLocation(int)<br/>Opposite(Boolean)<br/>Places([int])<br/>OpenClose</p> <p><b>Returns:</b> _</p> <p><b>Called From:</b> _</p> <p><b>Calls:</b> _</p>         | <p>1. Creates the ThisPlace object that has the properties of the default Place object</p> <p>2. Sets ThisPlace to be equal to the location given by IndexOfCurrentLocation</p> <p>3. The values of Direction and Opposite are checked to find the direction reference that needs to be changed. If Opposite is <i>False</i>, then the direction reference of ThisPlace for the direction Direction; set to NewLocationReference, but if Opposite is <i>True</i>, then the direction reference of ThisPlace for the opposite direction to Direction is set to NewLocationReference</p> <p>4. The location given by IndexOfCurrentLocation is set to be equal to ThisPlace</p> <p>5. Returns Places</p> |

**COPYRIGHT  
PROTECTED**



INSPECTION COPY

| Subroutine                                                                                                | Data                                                                                                                                                                                                                                                                                                                                                                                           | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-----------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| OpenClose (F)<br>      | Parameters:<br>Open (Boolean)<br>Items ([Item])<br>Places ([Place])<br>ItemToOpenClose (string)<br>CurrentLocation (int)<br>DirectionChange (int)<br>Items ([Item])<br>Places ([Place])<br>PlayGame<br>GetPositionOfCommand<br>GetResultForCommand<br>ExtractResultForCommand<br>ChangeStatusOfItem<br>GetIndexOfItem<br>ChangeLocationReference<br><br>Returns:<br><br>Called From:<br>Calls: | <ol style="list-style-type: none"> <li>If Open is <i>True</i>, creates the variable <i>Count</i> and sets it to "open"</li> <li>If Open is <i>False</i>, creates the variable <i>Count</i> and sets it to "close"</li> <li>Loops through each item in <i>Items</i> and if the status of the item is equal to <i>ItemToOpenClose</i>, the item is in <i>CurrentLocation</i> and the string of that item's commands is at least 4 characters long, checks the status of the item</li> <li>If the status of the item is equal to <i>Count</i>, returns -2, <i>Items</i> and <i>Places</i></li> <li>If the status of the item is "locked", returns -3, <i>Items</i> and <i>Places</i></li> <li>Otherwise, creates the variable <i>Position</i> and sets it to be equal to the position of <i>Count</i> in the door's list of commands, creates the variable <i>ResultForCommand</i> and sets it to be equal to the result of using <i>Command</i> on the door</li> <li>Changes the status of the door</li> <li>Sets <i>ActionWorked</i> to <i>True</i></li> <li>Loops through each place to change the appropriate location references in <i>CurrentLocation</i> and the location that is linked to <i>CurrentLocation</i> by the door</li> <li>Changes the status of the other side of the door</li> <li>If <i>ActionWorked</i> is <i>False</i>, returns -1, <i>Items</i> and <i>Places</i></li> <li>If <i>ActionWorked</i> is <i>True</i>, returns the integer value of <i>DirectionChange</i>, <i>Items</i> and <i>Places</i></li> </ol> |
| GetIndexOfItem (F)<br> | Parameters:<br>ItemNameToGet (string)<br>ItemIDToGet (int)<br>Items ([Item])<br>Count (int)<br>Called From:<br>Returns:                                                                                                                                                                                                                                                                        | <ol style="list-style-type: none"> <li>Creates the variable <i>Count</i> and sets it to 0, and creates the variable <i>StopLoop</i> and sets it to <i>False</i></li> <li>Loops through each item in <i>Items</i> by increasing the <i>Count</i> variable for as long as <i>Count</i> is less than the length of <i>Items</i> and <i>StopLoop</i> is <i>False</i></li> </ol>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

COPYRIGHT  
PROTECTED



INSPECTION COPY

| Subroutine             | Data                                                                                                                                                                                          | Description                                                                                                                                                                                                                                                                                                                    |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ChangeLocationOfItem ⑥ | Parameters: Items ([Item])<br>IndexOfItem (int)<br>NewLocation (int)<br>Returns: Items ([Item])<br>Called From: GetItem<br>TakeItemFromOtherCharacter<br>TakeRandomItemFromPlayer<br>Calls: - | 1. Creates the ThisItem object that has the properties of the default Item object<br>2. Sets ThisItem to be equal to the given item twice. Doing this the second time provides no benefit<br>3. Sets the location of ThisItem to be equal to NewLocation<br>4. Sets the given item to be equal to ThisItem<br>5. Returns Items |
| ChangeStatusOfItem ⑥   | Parameters: Items ([Item])<br>IndexOfItem (int)<br>NewStatus (string)<br>Returns: Items ([Item])<br>Called From: ChangeStatusOfDoor<br>OpenClose<br>Calls: -                                  | 1. Creates ThisItem object that has the properties of the default Item object<br>2. Sets ThisItem to be equal to the given item<br>3. Sets the status of ThisItem to be equal to NewStatus<br>4. Sets the given item to be equal to ThisItem<br>5. Returns Items                                                               |
| GetRandomNumber ⑥      | Parameters: LowerLimitValue (int)<br>UpperLimitValue (int)<br>Returns: (int)<br>Called From: RollDie<br>TakeRandomItemFromPlayer<br>Calls: -                                                  | 1. Returns random integer number between LowerLimitValue and UpperLimitValue                                                                                                                                                                                                                                                   |
| RollDie ⑥              | Parameters: Lower (string)<br>Upper (string)<br>Returns: (int)<br>Called From: UseItem                                                                                                        | 1. Creates LowerLimitValue variable, and sets it to 0<br>2. If the Lower string is made up entirely of numeric digits, sets LowerLimitValue to the integer value of Lower                                                                                                                                                      |

**COPYRIGHT  
PROTECTED**



INSPECTION COPY

| Subroutine             | Data                                                                                                                                                                                                                                                                       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ChangeStatusOfDoor (E) | Parameters:<br>Items ([Item])<br>CurrentLocation (int)<br>IndexOfItemToLockUnlock (int)<br>IndexOfOtherSideItemToLockUnlock (int)<br>Items ([Item])<br>UseItem<br>ChangeStatusOfItem<br>Say<br>Returns:<br>Called From:<br>Calls:                                          | <ol style="list-style-type: none"> <li>1. Checks whether the given door or the other side of the given door is in CurrentLocation</li> <li>2. If either is <i>True</i> and the status of the door is "locked", then the statuses of the door and the other side of the door are set to "close", and a message is displayed to say that the door has been unlocked</li> <li>3. If the status of the door is "close", then the statuses of the door and the other side of the door are set to "locked" and a message is displayed to say that the item has been locked</li> <li>4. Otherwise, displays a message to say that the item is open and so cannot be locked</li> <li>5. If neither the door nor the other side of the door is in CurrentLocation, displays a message to say that the key cannot be used in that location</li> </ol> |
| UseItem (E)            | Parameters:<br>Items ([Item])<br>ItemToUse (string)<br>CurrentLocation (int)<br>Places ([Place])<br>StopGame (Boolean)<br>Items ([Item])<br>PlayGame<br>Called From:<br>Calls:<br>ExtractResultForCommand<br>GetResultForCommand<br>GetPositionOfCommand<br>GetIndexOfItem | <ol style="list-style-type: none"> <li>1. Gets the index of the given item</li> <li>2. If the index of the item can be found, then checks whether the item is in INVENTORY or is in CurrentLocation, and has the status "usable"</li> <li>3. If either is <i>True</i>, gets the position of "use" in the item's list of commands and gets the result of using the "use" command on that item</li> <li>4. If SubCommand of the result is "say", then displays the SubCommandParameter of the result</li> <li>5. If SubCommand of the result is "lockunlock", then changes the status of the relevant doors</li> </ol>                                                                                                                                                                                                                        |

**COPYRIGHT  
PROTECTED**



INSPECTION COPY

| Subroutine | Data                                                                                                                                                                                                                                             | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ReadItem @ | Parameters:<br>Items ([Item])<br>ItemToRead (string)<br>CurrentLocation (int)<br>-<br>PlayGame<br>GetIndexOfItem<br>GetPositionOfCommand<br>GetResultForCommand<br>ExtractResultForCommand                                                       | 1. Gets the index of the given item<br>2. If the index of the item couldn't be found, then displays a message to say that the item cannot be found<br>3. If the item doesn't have "read" in its list of commands, then displays a message to say that the item cannot be read<br>4. If the item is not in CurrentLocation or INVENTORY, then displays a message to say that the item cannot be found<br>5. Otherwise, gets the position of "read" in the item's list of commands and gets the result of executing the "read" command on that item<br>6. If the command of the result is "say", then displays the SubCommandParameter of the result                                                                                                                                                            |
| GetItem @  | Parameters:<br>Items ([Item])<br>ItemToGet (string)<br>CurrentLocation (int)<br>(Boolean)<br>Items ([Item])<br>PlayGame<br>-<br>ExtractResultForCommand<br>GetResultForCommand<br>GetPositionOfCommand<br>GetIndexOfItem<br>ChangeLocationOfItem | 1. Gets the index of the given item<br>2. If the index of the item couldn't be found, then displays a message to say that the item cannot be found<br>3. If the item is in INVENTORY, displays a message to say that the item is already in the inventory<br>4. If the item doesn't have the command "get", then displays a message to say that the player can't get the item<br>5. If the item is stored in another item and the location of the item that contains it is not CurrentLocation, displays a message to say that the item cannot be found<br>6. If the item is not stored in another item and its location is not CurrentLocation, displays a message to say that the item cannot be found<br>7. If the item was found and has "get" in its list of commands, gets the position of "get" in the |

COPYRIGHT  
PROTECTED



INSPECTION COPY

| Subroutine                                                                                                            | Data                                                                                                                                                                                                                                                                                        | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CheckIfDiceGamePossible (F)<br>    | <b>Parameters:</b> Items ([Item])<br>Characters ([Character])<br>OtherCharacterName (string)<br>(Boolean)<br><b>Returns:</b> IndexOfPlayerDie (int)<br>IndexOfOtherCharacterDie (int)<br>IndexOfOtherCharacterDie (int)<br><b>Called From:</b> PlayDiceGame<br><b>Calls:</b> GetIndexOfItem | <ol style="list-style-type: none"> <li>Loops through each item in Items and if an item is in INVENTORY and has "die" in its name, gets the index of that die and sets PlayerHasDie to <i>True</i></li> <li>Loops through each character in Characters until a character is found in the same location as the player and the name OtherCharacterName or all of the characters have been checked</li> <li>If a character is found that has the name OtherCharacterName and is in the same location as the player, sets PlayerIsInSameRoom to <i>True</i> and loops through each item in Items</li> <li>If an item is held by the other character and has "die" in its name, gets the index of that die and sets OtherCharacterHasDie to <i>True</i></li> <li>Returns result of (PlayerHasDie and PlayerIsInSameRoom and OtherCharacterHasDie), IndexOfPlayerDie, IndexOfOtherCharacter and IndexOfOtherCharacterDie</li> </ol> |
| TakeItemFromOtherCharacter (F)<br> | <b>Parameters:</b> Items ([Item])<br>OtherCharacterID (int)<br><b>Returns:</b> Items ([Item])<br><b>Called From:</b> PlayDiceGame<br><b>Calls:</b> ChangeLocationOfItem                                                                                                                     | <ol style="list-style-type: none"> <li>Loops through all of the items in Items, and checks whether each item is held by the given character</li> <li>If the item is held by that character, the name of that item is added to a list. The message "Which item do you want to take? They have:" is displayed, followed by each item in the list, with each item separated by "," and the list ending with "." , although the last item in the list is not displayed unless it is the only item in the list</li> <li>Gets the result of user input</li> <li>If the user inputs the name of an item in the list, displays a message to say that the player now has that item and the item is moved to the player's inventory</li> <li>Otherwise, displays a message to say that the character doesn't have that item</li> </ol>                                                                                                 |

COPYRIGHT  
PROTECTED



INSPECTION COPY

| Subroutine Name                                                                                       | Data                                                                                                                                                                                                                                                                                                                                    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PlayDiceGame ©<br> | <b>Parameters:</b><br>Characters ([Character])<br>Items ([Item])<br>OtherCharacterName (string)<br>Items ([Item])<br>PlayGame<br><b>Returns:</b><br>CheckIfDiceGamePossible<br><b>Called From:</b><br>GetPositionOfCommand<br><b>Calls:</b><br>GetResultForCommand<br>RollDie<br>TakeItemFromOtherCharacter<br>TakeRandomItemFromPlayer | <ol style="list-style-type: none"> <li>1. Calls CheckIfDiceGamePossible to see if a dice game can be played</li> <li>2. If a dice game can't be played, displays the message "You can't play a dice game."</li> <li>3. If a dice game can be played, gets the position of the "use" in the player's die's list of commands and gets the result of using the command on that die to get the result of the dice roll</li> <li>4. Displays a message to show what the player roll</li> <li>5. Gets the position of "use" in the other character's list of commands and gets the result of using the "use" command on that die to get the result of the dice roll</li> <li>6. Displays a message to show what the other character rolled</li> <li>7. If the player's roll was higher than the other character's, display a message to say that the player has won, and allow them to take an item from that character</li> <li>8. If the player's roll was lower than the other character's, display a message to say that the player has lost, and give one of their items to the other character</li> <li>9. If the player's roll was the same as the other character's, display a message to say that the game was a draw</li> <li>10. return Items</li> </ol> |
| MoveItem ©                                                                                            | <b>Parameter:</b><br>Items ([Item])<br>ItemToMove (string)<br>CurrentLocation (int)<br><b>Returns:</b><br>-<br><b>Called From:</b><br>PlayGame<br><b>Calls:</b><br>GetIndexOfItem<br>GetPositionOfCommand<br>GetResultForCommand                                                                                                        | <ol style="list-style-type: none"> <li>1. Gets the index of the given item</li> <li>2. If the index of the item is found, then checks the length of the item's list of commands</li> <li>3. If the list of commands has at least 4 characters and contains "move", gets the position of "move" in the item's list of commands and gets the result of using the "move" command on that item</li> <li>4. If SubCommand of the result is "say", then displays the SubCommandParameter of</li> </ol>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

**COPYRIGHT  
PROTECTED**



INSPECTION COPY

| Subroutine                      | Data                                                                                                              | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------|-------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DisplayInventory®               | Parameters: Items ([Item])<br>Returns: -<br>Called From: Examine<br>Calls: -                                      | 1. Prints empty line<br>2. Prints "You are currently carrying the following items:"<br>3. Loops through every item in Items and if the location of that item is INVENTORY, prints the name of that item                                                                                                                                                                                                                                                                                                                               |
| DisplayGettableItemsInLocation® | Parameters: Items ([Item])<br>CurrentLocation (int)<br>Returns: -<br>Called From: PlayGame<br>Calls: -            | 1. Loops through each item in Items and checks whether the item is in CurrentLocation and whether its status contains "gettable"<br>2. If both of these conditions are True, appends the name of the item to a string starting "On the floor there is:"<br>3. If the name of another item has already been added to the string, prints a " , " before the name of the next item<br>If the name of any item was appended to the string, a " " is appended to the end                                                                   |
| DisplayOpenCloseMessage®        | Parameters: ResultOfOpenClose (int)<br>OpenCommand (Boolean)<br>Returns: -<br>Called From: PlayGame<br>Calls: Say | 1. If ResultOfOpenClose is greater than 0, then checks whether the door has been opened, otherwise displays a message to say that the door has been closed<br>2. If ResultOfOpenClose is -3, displays a message to say that the door is locked<br>3. If ResultOfOpenClose is -2, displays a message to say that the door is already in that state<br>4. If ResultOfOpenClose is -1, displays a message to say that the item can't be opened<br>5. If ResultOfOpenClose is -1, displays a message to say that the item can't be opened |

COPYRIGHT  
PROTECTED



INSPECTION COPY

| Subroutine                                                                                        | Data                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PlayGame ©<br> | <div> <div>Parameters:</div> <div>           Characters ([Character])<br/>           Items ([Item])<br/>           Places ([Place])<br/>           -<br/>           Main<br/>           DisplayGettableItemsInLocation<br/>           GetInstruction<br/>           ExtractCommand<br/>           GetItem<br/>           UseItem<br/>           Go<br/>           ReadItem<br/>           Examine<br/>           OpenClose<br/>           DisplayOpenCloseMessage<br/>           MoveItem<br/>           PlayDiceGame<br/>           Say         </div> </div> <div> <div>            Returns:<br/>           Called From:<br/>           Calls:         </div> </div> | <ol style="list-style-type: none"> <li>If the game has ended, waits for user input and then terminates, otherwise the subroutine runs the following loop</li> <li>If the player has moved to a new location prints two empty lines, the description of the location and the given items at that location</li> <li>Gets input from the user, converts it to a Command and splits it into a Command and an Instruction</li> <li>If Command is "get", tries to get the given item</li> <li>If Command is "use", tries to use the given item</li> <li>If Command is "go", tries to move the character in the given direction</li> <li>If Command is "read", tries to read the given item</li> <li>If Command is "examine", tries to examine the given item, character or inventory</li> <li>If Command is "open", tries to open the given item</li> <li>If Command is "close", tries to close the given item</li> <li>If Command is "move", tries to move the given item</li> <li>If Command is "say", calls displays the Instruction to the input</li> <li>If Command is "playdice", tries to play a dice game with the given character</li> <li>If Command is "quit", displays a message to say that the player has given up and ends the game</li> <li>If Command is anything else, displays a message to say that the command isn't recognised</li> </ol> |

COPYRIGHT  
PROTECTED



INSPECTION COPY

| Subroutine | Data                                                                                                                                                                                                                                                                                                                 | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LoadGame ⑥ | <div>Parameters:</div> <div>Filename (string)</div> <div>Characters ([Character])</div> <div>Items ([Item])</div> <div>Places ([Place])</div> <div>(Boolean)</div> <div>Characters ([Character])</div> <div>Items ([Item])</div> <div>Places ([Place])</div> <div>Called From: Main</div> <div>Created From: -</div> | <ol style="list-style-type: none"> <li>1. If an exception is thrown, returns <i>False</i>, Characters, Items and Places, otherwise the subroutine runs as follows</li> <li>2. Creates the a file object <i>f</i> to read the contents of the given file</li> <li>3. Reads the number of characters from the file</li> <li>4. Creates <i>TempCharacter</i>, sets its properties to the values read from the file and adds this character to <i>Characters</i>. This is repeated for each character.</li> <li>5. Reads the number of places from the file</li> <li>6. Creates <i>TempPlace</i>, sets its properties to the values read from the file and adds this place to <i>Places</i>. This is repeated for each place.</li> <li>7. Reads the number of items from the file</li> <li>8. Creates <i>TempItem</i>, sets its properties to the values read from the file and adds this item to <i>Items</i>. This is repeated for each item.</li> <li>9. Returns <i>True</i>, Characters, Items and Places</li> </ol> |
| Main ⑥     | <div>Parameters:</div> <div>-</div> <div>Returns:</div> <div>-</div> <div>Called From:</div> <div>-</div> <div>Calls:</div> <div>LoadGame</div> <div>PlayGame</div>                                                                                                                                                  | <ol style="list-style-type: none"> <li>1. Creates <i>Items</i>, <i>Characters</i> and <i>Places</i> variables as empty lists</li> <li>2. Gets filename from user input (the program adds ".gm" to the user's input, so the file will not be recognised if the user enters the .gm to the end of the filename themselves)</li> <li>3. Prints an empty line</li> <li>4. Tries to load the game from the given file</li> <li>5. If the game is successfully loaded, calls <i>PlayGame</i></li> <li>6. If the game is not successfully loaded, displays a message to say that the game could not be loaded and terminates the program after the user enters input</li> </ol>                                                                                                                                                                                                                                                                                                                                             |

COPYRIGHT  
PROTECTED



INSPECTION COPY

# Program Classes

The classes defined in the program, and their attributes, are described below.

| Class     | Description                                                        |
|-----------|--------------------------------------------------------------------|
| Place     | Class that defines the properties of the game's locations          |
| Character | Class that defines the properties of each of the game's characters |
| Item      | Class that defines the properties of each of the game's items      |

# Place Attributes

| Attribute   | Type    | Default Value | Description                                                                                                          |
|-------------|---------|---------------|----------------------------------------------------------------------------------------------------------------------|
| Description | String  | ""            | The text description of the place, which is shown to the player whenever they move to this place                     |
| ID          | Integer | 0             | The ID of this place, which allows it to be referenced by the program or other objects                               |
| North       | Integer | 0             | The ID of the location to the north of this place; if this is 0, then there is no location to the north of the place |
| East        | Integer | 0             | The ID of the location to the east of this place; if this is 0, then there is no location to the east of the place   |
| South       | Integer | 0             | The ID of the location to the south of this place; if this is 0, then there is no location to the south of the place |
| West        | Integer | 0             | The ID of the location to the west of this place; if this is 0, then there is no location to the west of the place   |
| Up          | Integer | 0             | The ID of the location above this place; if this is 0, then there is no location above the place                     |
| Down        | Integer | 0             | The ID of the location below this place; if this is 0, then there is no location below the place                     |

# Item Attributes

| Attribute   | Type    | Default Value | Description                                                                                                                  |
|-------------|---------|---------------|------------------------------------------------------------------------------------------------------------------------------|
| ID          | Integer | 0             | The ID of this item, which allows it to be referenced by the program or other objects.                                       |
| Location    | Integer | 0             | The ID of the place, item, or character holding this item; if this is 0, then the item is held by a place, item or character |
| Description | String  | ""            | The text description of the item, which is shown to the player whenever they examine this item                               |
| Status      | String  | ""            | A string containing various attributes of an item (e.g. if a door has the status "locked", it cannot be opened)              |
| Name        | String  | ""            | The name of the item, which is displayed to the player and used by the player to interact with this item                     |
| Commands    | String  | ""            | A string containing the commands that can be used on this item (such as "get", "use" and "drop")                             |
| Results     | String  | ""            | A string containing the results of using commands on this item (the default result moves the item to INVENTORY)              |

COPYRIGHT  
PROTECTED



INSPECTION COPY



INSPECTION COPY

INSPECTION COPY

INSPECTION COPY

Get Item

Say

Speech

Subcommand, Subcommandparameter, ResultForCommand

Subcommand, Subcommandparameter

GetResult

Items, IndexOfItem, NewLocation

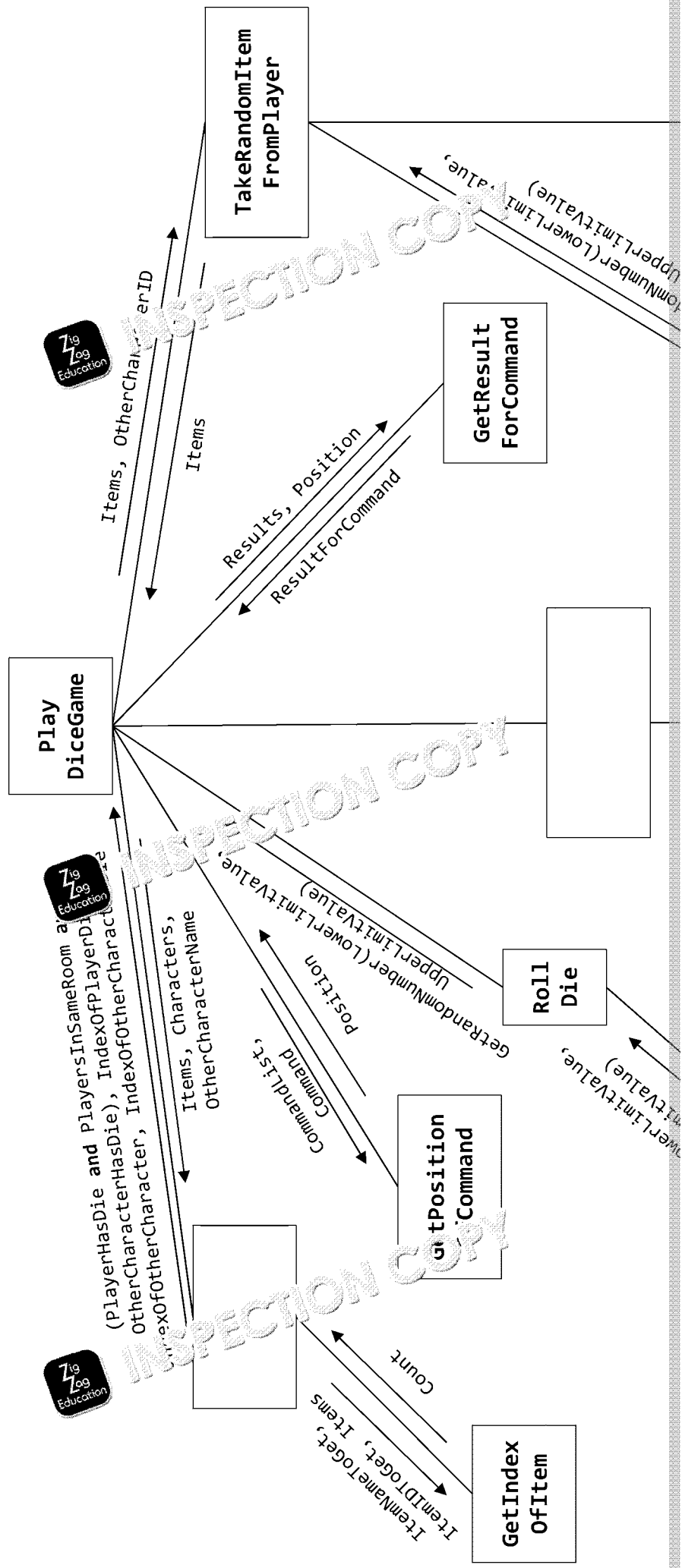
Items

Count  
ItemNameToGet,  
ItemIDToGet, Items

COPYRIGHT  
PROTECTED



INSPECTION COPY



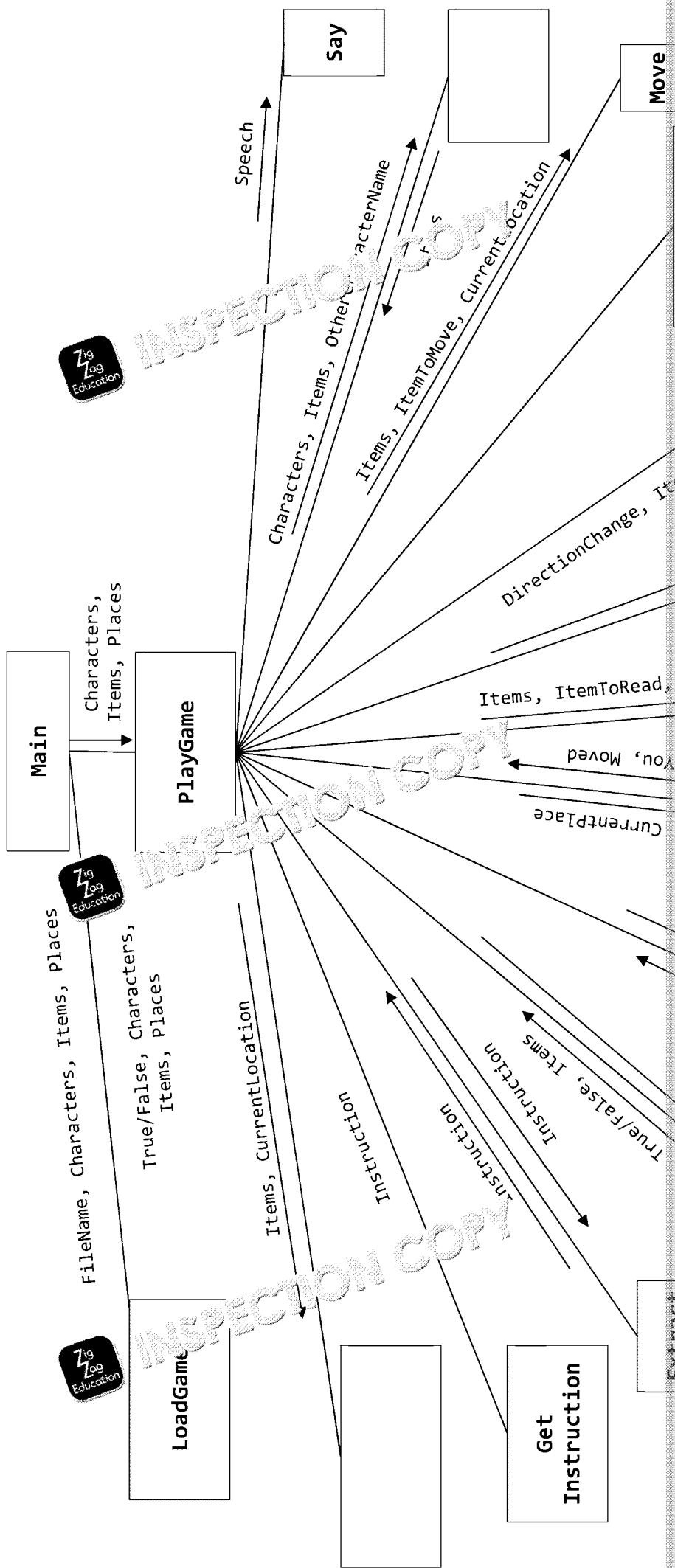
COPYRIGHT  
PROTECTED



INSPECTION COPY

## Structure Diagram Activity 3 (Main)

## AQA Text Adventures



COPYRIGHT  
PROTECTED



INSPECTION COPY

# AQA Text Adventures

## Written Questions (Python)

These questions refer to the preliminary material and require you to load the skeleton program and make any additional programming.

1. State the name of an identifier for:
  - a) A local variable in the function RollDie [1]
  - b) A function that returns multiple variables [1]
  - c) Four built-in functions used in the function TakeItemFromOtherCharacter [1]
  - d) A class that has exactly four attributes [1]
  - e) A constructor [1]
  - f) A variable used as a constant [1]
  - g) Two user-defined functions with four parameters [2]
  - h) A user-defined function, other than Main, that takes no arguments [1]
2. Explain the purpose of the following line in the GetInstruction subroutine:

```
Instruction = raw_input("> ").lower()
```
3. Explain the role of the variable Moved in the function Go. [3]
4. Explain the purpose of the command break used in the function GetResult. [1]
5. Using a comment from the Skeleton Program, describe what it meant by the phrase 'roll the dice'. [1]
6. Describe in detail the role of the procedure DisplayInventory. [5]
7. Describe the role of the variable TempCharacter in the function LoadGame. [1]
8. Explain why the LowerLimitValue in the function RollDie is initialised to 1. [1]
9. Explain the purpose of each of the two while loops in the function ExtractCharacter. [2]
10. Describe the purpose of the first for loop in the function CheckIfDiceGame. [1]
11. Describe the changes that would need to be made so that the game would output 'nothing' if the player attempts to examine an empty inventory. [4]
12. Describe the changes that would need to be made to the function GetInstruction so that it returns the instruction if two or more characters has been entered before the instruction returns. [2]
13. Describe the effect of a file not being found during the function LoadGame. [1]

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



- END OF TEST -

# AQA Text Adventures

## Written Questions (Python)

These questions refer to the preliminary material and require you to load the skeleton code. They do not require any additional programming.

1. State the name of an identifier for:

a) A local variable in the function `RollDie` [1]

.....

b) A function that returns multiple variables [1]

.....

c) Four built-in functions used in the function `TakeItemFromOtherCharacter`

.....

.....

d) A class that has exactly four attributes [1]

.....

e) A constructor [1]

.....

f) A variable with a constant value [1]

.....

g) Two user-defined functions with four parameters [2]

.....

h) A user-defined function, other than `Main`, that takes no arguments [1]

.....

2. Explain the purpose of the following line in the `GetInstruction` subroutine

```
Instruction = raw_input("> ").lower()
```

.....

.....

.....

.....

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



3. Explain the role of the variable `Moved` in the function `Go`. [3]

.....

.....

.....

.....

4. Explain the purpose of the comparison `b == 0` used in the function `GetResult`. [3]

.....

.....

.....

.....

5. Using any example from the Skeleton Program, describe what it meant by the phrase "a character is not in the inventory". [3]

.....

.....

.....

.....

6. Describe in detail the role of the variable `Inventory` in the function `DisplayInventory`. [5]

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

7. Describe the role of the variable `TempCharacter` in the function `LoadGame`. [3]

.....

.....

.....

.....

.....

**COPYRIGHT  
PROTECTED**



8. Explain why the `LowerLimitValue` in the function `RollDie` is initialised

.....

.....

.....

.....

.....

9. Explain the purpose of each of the two while loops in the function `ExtractCharacter`.  
While loop 1

.....

.....

.....

.....

While loop 2

.....

.....

.....

10. Describe the purpose of the first `for` loop in the function `CheckIfDiceGame`.

.....

.....

.....

.....

.....

.....

11. Describe the changes that would need to be made so that the game would output 'nothing' if the player attempts to examine an item in their inventory. [4]

.....

.....

.....

.....

.....

**COPYRIGHT  
PROTECTED**



12. Describe the changes that would need to be made to the function `GetInstruction` if two or more characters has been entered before `GetInstruction` returns.



INSPECTION COPY

13. Describe the effect of a file not being found during the function `LoadGame`.



INSPECTION COPY

- END OF TEST -

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



INSPECTION COPY



# AQA Text Adventures

## Programming Tasks (Python)

The following require you to open the skeleton program and make modifications

### Task 1

This question refers to `PlayGame` as well as a new subroutine `ShowHelp`.

Currently, users are not offered any help regarding valid commands. If a user enters a command, they are told 'Sorry, you don't know how to...'

Add a new subroutine called `ShowHelp`, which should display the following text:

**You can enter one of the following commands:**

**go, get, use, examine, say, quit, read, move, open, close and playdice.**

Modify the subroutine `PlayGame` so that if a user enters the command 'help', the

#### Evidence you need to provide:

- Your amended SOURCE CODE for `PlayGame`
- Your SOURCE CODE for the new `ShowHelp` subroutine
- One screenshot showing the output that results from the following:
  - Run the program and load the file 'flag1'
  - Type 'help' at the first prompt

### Task 2

This question refers to `Examine` and `PlayGame`.

Currently, users are able to examine their inventory, as well as objects within a room.

Modify `Examine` so that if the user has entered the text 'examine room', the description of the room should be re-displayed. A call to `DisplayGettableItemsInLocation` should also be made.

You should pass `Places` from `PlayGame` to `Examine`.

#### Evidence you need to provide:

- Your amended SOURCE CODE for `Examine`
- Your amended SOURCE CODE for `PlayGame`
- One screenshot showing the output that results from the following:
  - Run the program and load the file 'flag1'
  - Type 'examine room' at the first prompt

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



### Task 3

This question refers to `PlayGame`.

Currently, if the user types 'quit' at the prompt, the game ends without any further prompts.

Modify `PlayGame` so that an entry of 'quit' triggers an output of **'Are you sure?'**

Entering either 'y' or 'yes' in any combination of upper or lowercase should terminate the game.

**'You decide to give up, try again another time.'** should be displayed.

Any other input should return the player to the game and the main prompt.

#### Evidence you need to provide:

- Your amended SOURCE CODE for `PlayGame`
- One screenshot showing the output that results from the following:
  - Run the program and load the file 'flag1'
  - Type 'quit' at the first prompt
  - Type 'no' in response to 'Are you sure'
  - Type 'quit' at the next prompt
  - Type 'yes' in response to 'Are you sure'
  - Press enter after resulting output

### Task 4

This question refers to `RollDie`.

Currently, if you enter a non-integer value when prompted for a minimum or maximum, the program will throw an error.

Modify `RollDie` so that if the user enters a non-integer value, the error is handled gracefully. If they enter an integer from 1 to 6, the program should proceed to the next prompt.

#### Evidence you need to provide:

- Your amended SOURCE CODE for `RollDie`
- One screenshot showing the output that results from the following:
  - Run the program and load the file 'flag2'
  - Type 'playdice guard' at the first prompt
  - Type 'one' at the 'Enter minimum: ' prompt
  - Type '1.0' at the next 'Enter minimum: ' prompt
  - Type '1' at the next 'Enter minimum: ' prompt
  - Type 'six' at the 'Enter maximum: ' prompt
  - Type '6.0' at the next 'Enter maximum: ' prompt
  - Type '6' at the next 'Enter maximum: ' prompt

**COPYRIGHT  
PROTECTED**



## Task 5

This question refers to `DisplayInventory`.

Currently, when the inventory is examined items are displayed in the order in which they are stored in the binary file.

Modify the `DisplayInventory` subroutine, so that it uses a bubble sort to sort the items into alphabetical order based on their `Name` attribute before displaying the list of items. You should write the sorting algorithm yourself, and should not use any library sorting functions.

### Evidence you need to provide:

- Your amended SOURCE CODE for `DisplayInventory`
- One screenshot showing the output that results from the following:
  - Run the program and load the file 'flag1'
  - Type 'examine inventory' at the first prompt
  - Type 'get red die' at the second prompt
  - Type 'examine inventory' at the third prompt

## Task 6

This question refers to `GetItem` and `TakeItemFromOtherCharacter` as well as `MAX_INVENTORY`.

Currently, the player's inventory can contain as many items as can be found.

Create an integer constant called `MAX_INVENTORY` and set it to 4. When an attempt is made to add an item to a player's inventory with four or more items already in it, the message 'Inventory full' should be displayed. The check for a full inventory should take place before any existing selection state messages should be output. This check should be within `GetItem` other than 'Inventory full'.

This check should also be applied to `TakeItemFromOtherCharacter` so that if, after winning a dice game, the message 'Inventory full' will be displayed if the player's inventory is full.

### Evidence you need to provide:

- Your amended SOURCE CODE for `GetItem`
- Your amended SOURCE CODE for `TakeItemFromOtherCharacter`
- Your SOURCE CODE for the new `MAX_INVENTORY` constant
- One screenshot showing the output that results from the following:
  - Run the program and load the file 'flag1'
  - Type 'get torch' at the first subsequent prompt
  - Type 'get red die' at the second prompt
  - Type 'examine inventory' at the third prompt
- One screenshot showing the output that results from the following:
  - Run the program and load the file 'flag2'
  - Type 'playdice game' at the first subsequent prompt
  - At each of the next two prompts, type '6' (repeat the previous step)
  - Type 'examine inventory' at the next prompt

**COPYRIGHT  
PROTECTED**



## Task 7

This question refers to `PlayGame` as well as a new subroutine `DropItem`.

Currently, items can be picked up but not dropped.

Create a new subroutine called `DropItem` which accepts three parameters:

- A variable called 'Items', which will contain all items in the game
- A variable called 'ItemToDrop', which will be the name of the item leaving
- A variable called 'CurrentLocation', which will be the ID of the player's location

`DropItem` should check that the player is carrying the item. If they are not, the item should be displayed (with the item name displayed in place of \_\_\_\_). Otherwise, the room with the text '\_\_\_\_ **dropped**' indicating which item was dropped.

You should modify `PlayGame` to call `DropItem` if a command of 'drop' is entered.

### Evidence you need to provide:

- Your amended SOURCE CODE for `PlayGame`
- Your SOURCE CODE for the new `DropItem` subroutine
- One screenshot showing the output that results from the following:
  - Run the program and load the file 'flag1'
  - Type 'drop flask' at the first prompt
  - Type 'drop flask' again at the second prompt
  - Type 'examine inventory' at the third prompt

## Task 8

This question refers to `Main`.

Currently, if the user enters the name of a non-existent file, the program ends with a different file name. Additionally, the file extension '.gme' is always appended to the file name. If the user enters a file name that does not include the extension themselves. This would result in a file not being found.

Modify the user prompt in `Main` to read 'Enter filename or enter Q to quit'. If the user enters 'Q', the program should end. If the user enters a valid file name, the file should load as normal. Otherwise, the program should continually re-display until the user has entered 'Q' or a file name that successfully loads.

If the user enters a file name that contains the string '.gme' the program should not append the extension.

You should only make changes to `Main`.

### Evidence you need to provide:

- Your amended SOURCE CODE for `Main`
- One screenshot showing the output that results from the following:
  - Type 'flag3' when prompted for a file name
  - At the next prompt, type 'flag2.gme'

**COPYRIGHT  
PROTECTED**



## Task 9

This question relates to `PlayGame` as well as a new subroutine `DisplayCharacters`.

In order for a developer to work effectively on existing code, it is helpful for them to know the content of a data structure. Create a new subroutine called `DisplayCharacters` with a single parameter in the form of `Characters`.

When `DisplayCharacters` is called, it should display four column headers in the following order:

- ID (followed by three spaces)
- NAME (followed by eight spaces)
- DESCRIPTION (followed by 53 spaces)
- CURRENTLOCATION

Beneath these columns, all characters within the game should be displayed, with the attributes ID, Name, Description and CurrentLocation should each align with its respective column header.

Modify `PlayGame` so that if 'debugchar' is entered during gameplay, a call to `DisplayCharacters` is made.

### Evidence you need to provide:

- Your amended SOURCE CODE for `PlayGame`
- Your SOURCE CODE for the new `DisplayCharacters` subroutine
- One screenshot showing the output that results from the following:
  - Type 'flag1' when prompted for a file name
  - At the next prompt, enter 'debugchar'

INSPECTION COPY

COPYRIGHT  
PROTECTED



## Task 10

This question relates to `PlayGame` as well as a new subroutine `FillVessel`.

In the `flag1` game, there is a room below the starting location that contains a barrel. You need to write a new subroutine called `FillVessel` that requires the following parameters

- A variable called 'Characters', that stores all of the characters in the game
- A variable called 'Items', that stores all of the items in the game
- A variable called 'Places', that stores all of the places in the game
- A variable called 'Vessel', that stores the name of the item to be filled

`FillVessel` should check whether the player and the barrel are in the same location. If they are, it should check whether the barrel's `Status` attribute contains the text 'container' and does not contain the text 'filled'. If these conditions are met, the barrel should be in the player's inventory.

If any of these conditions are not met, the message '**You cannot do that**' should be displayed. The item's `Name` attribute should have the text ' of water' appended to it, and the text should be displayed, with the vessel appended to the end.

If the vessel's name attribute already contains the text 'of water', the text '**Already**' should be displayed.

Add a call to `FillVessel` in `PlayGame` that will occur if the player enters the command 'fill flask'.

### Evidence you need to provide:

- Your amended SOURCE CODE for `PlayGame`
- Your SOURCE CODE for the new `FillVessel` subroutine
- One screenshot showing the output of the following:
  - Type 'flag1' when prompted for a file name
  - At the next prompt, type 'open trapdoor'
  - At the next prompt, type 'go down'
  - At the next prompt, type 'fill flask'
  - At the next prompt, type 'fill apple'
  - At the next prompt, type 'fill flask of water'
  - At the next prompt, type 'examine inventory'

INSPECTION COPY

COPYRIGHT  
PROTECTED



INSPECTION COPY



## Task 11

This question relates to `Go`, `PlayGame` and a new class, `History`.

Create a new class called `History`, with two attributes:

- An variable called 'Pointer', that is initially set to -1
- A five-element list called 'Locations', that is initially filled with empty values

`History` will use a stack, to allow players to backtrack through previous locations. You should use an implementation of a stack using an array, which will be subject to two subroutines:

- `Push` should require two parameters called 'self' and 'NewLocation'. If the stack is full, it should be moved down by 1 element (so the content of element 0 is lost), and 'NewLocation' should be added to the top. If the stack is not full, the pointer attribute should be incremented and 'NewLocation' should be added to the element indicated by `Pointer`. The `Push` subroutine should return the value of `Pointer`.
- `Pop` requires only a parameter called 'self' and returns an integer. If the stack is empty, -1 should be returned. Otherwise the value in the element indicated by `Pointer` should be returned, and `Pointer` should be decremented.

In `PlayGame`, create an instance of the `History` class called 'past'. When the `go back` command, this instance should be passed to the `Go` subroutine as an additional parameter.

In `Go`, modify the parameters to accept the instance of the `History` class. As a player moves to a new location, add the location onto the stack before they move to a new one. If the instruction 'go back' is entered, use a call to the `Pop` subroutine of `History`. If the call to `Pop` returns -1, display the text 'You cannot go back'

**Evidence you need to provide:**

- Your amended SOURCE CODE for the new `History` class
- Your amended SOURCE CODE for `PlayGame`
- Your amended SOURCE CODE for `Go`
- One screenshot showing the output that results from the following:
  - Type 'flag2' when prompted for a file name
  - At the next prompt, type 'go east'
  - At the next prompt, type 'go back'
  - At the next prompt, type 'go back' a second time

INSPECTION COPY

**COPYRIGHT  
PROTECTED**




  
 ZSPECTRON COPY



- 



- 



## Task 14

This question relates to `CheckIfDiceGamePossible`.

Currently, if a dice game is not possible, the program outputs the message 'You can't play more than one of these conditions are met, only one should be displayed, with conditions which they are numbered above. So if the player has no die, the program will not output that the opponent has a die. If the opponent is not present, no check will be made.

Modify `CheckIfDiceGamePossible` so that the following messages are output:

1. If the player has no die, the output should read 'Player has no die'
2. If the player and the opponent are not in the same room, the output should read '\_\_\_\_ is not in the same room as \_\_\_\_' (with \_\_\_\_ being replaced by the name of the other character)
3. If the opponent has no die, the output should be '\_\_\_\_ has no die' (with \_\_\_\_ being replaced by the name of the other character)

In all cases, these messages should still be followed by the message 'You can't play more than one of these conditions are met, only one should be displayed, with conditions which they are numbered above. So if the player has no die, the program will not output that the opponent has a die. If the opponent is not present, no check will be made.

No code should be modified besides the content of `CheckIfDiceGamePossible`.

### Evidence you need to provide:

- Your amended SOURCE CODE for `CheckIfDiceGamePossible`
- One screenshot showing the output that results from the following:
  - Type 'flag1' when prompted for a file name
  - At the next prompt, type 'playdice guard'
  - At the next prompt, type 'get red die'
  - At the next prompt, type 'playdice guard'

## Task 15

This question relates to `DisplayInventory`.

Currently, when the user enters 'examine inventory', a list of the names of the items is displayed.

Modify `DisplayInventory` so that descriptions are displayed alongside the names of the items. The descriptions should be displayed in round brackets. Additionally, the opening brackets of the descriptions should be displayed regardless of variations in an item's name.

### Evidence you need to provide:

- Your amended SOURCE CODE for `DisplayInventory`
- One screenshot showing the output that results from the following:
  - Type 'flag1' when prompted for a file name
  - At the next prompt, type 'examine inventory'

**COPYRIGHT  
PROTECTED**



# AQA Text Adventures

## Programming Tasks - Extension (Python)

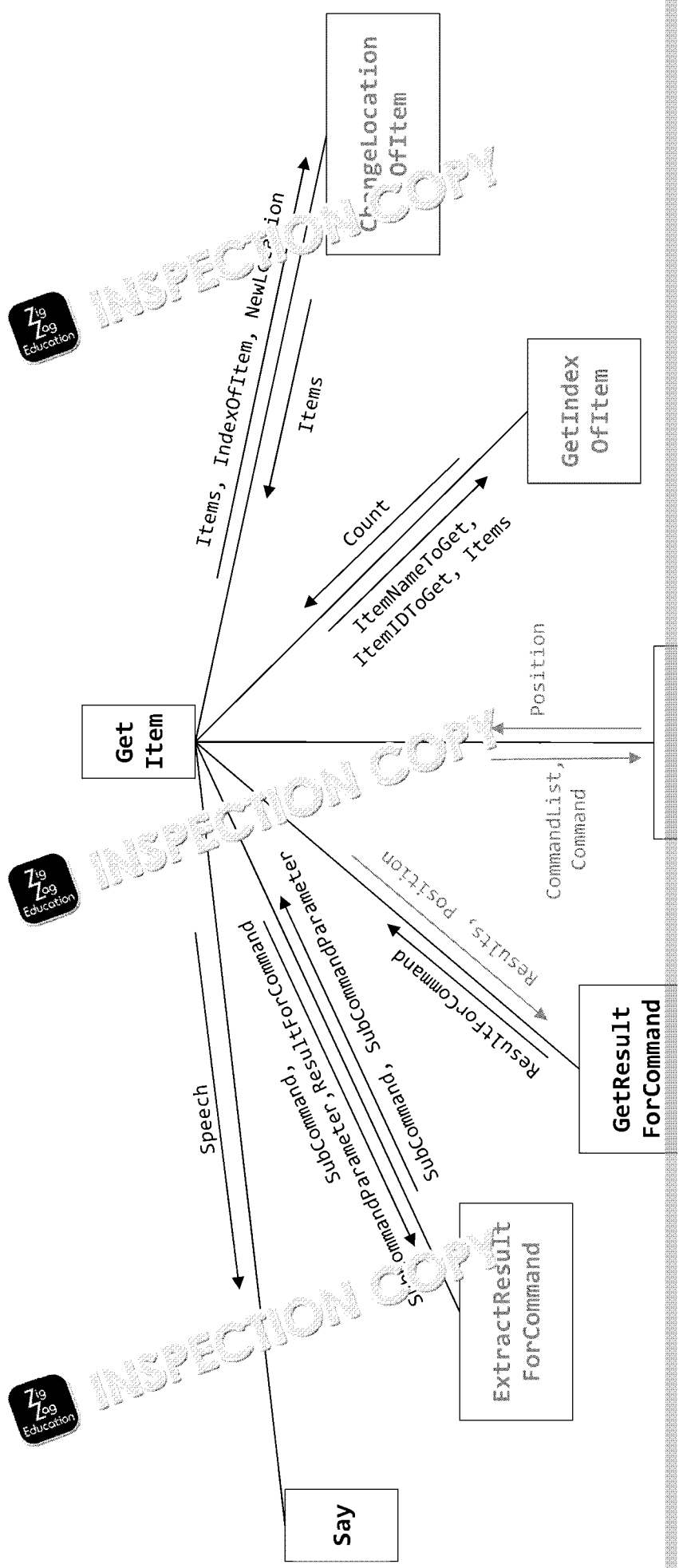
These challenges are presented without solutions, and offered to further explore a program. They are listed here in approximate order of difficulty, beginning with

1. Add a 'restart' command to begin the current game again from the start of a new game by entering a file name
2. Modify `UseItem` to incorporate a win condition that involves using an object and the flag
3. Randomise the location of the guard
4. In the original program, the guard's blue die is not presented as an option to take after winning a dice game unless it is the guard's only remaining item to take even if the guard has other items. Modify `TakeItemFromGuard` so that a player can only take the items that are listed after they have won a dice game
5. Modify `Go` so that the player will automatically try to unlock any locked doors in their way
6. Incorporate an additional command 'look', which could be used instead of 'go' to view a location, assuming 'go' wouldn't have been blocked by a closed door
7. Attempting to pick up an item by using part of its name (such as 'die' instead of 'blue die') successfully pick up the item
8. If the player is carrying two dice, both can be rolled and the total used against the guard
9. In the starting room for 'flag1.gme', the trapdoor is hidden by a rug but can be accessed by moving the rug. Edit the game so that the rug must be moved before the trapdoor can be accessed
10. Edit `PlayDiceGame` so that characters cannot lose a die in a dice game (a character can possess nothing but dice)
11. If you enter 'use truncheon' in the same room as the guard, the guard is killed and items can be taken without the need to play dice
12. Incorporate a 'save' command for players to save their progress, which could be used to save a game file
13. An option to create a game file from the console, saving the new game file to a specified location

INSPECTION COPY

**COPYRIGHT  
PROTECTED**

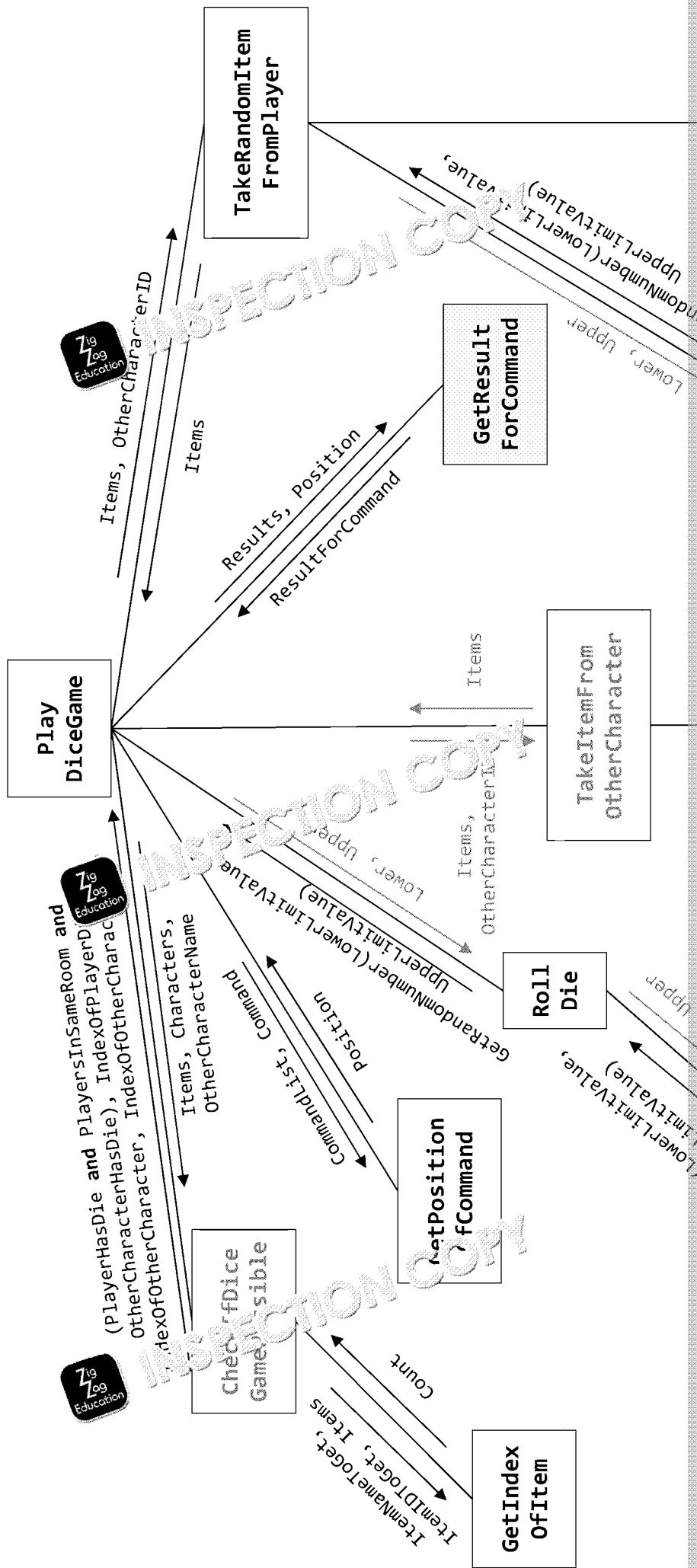


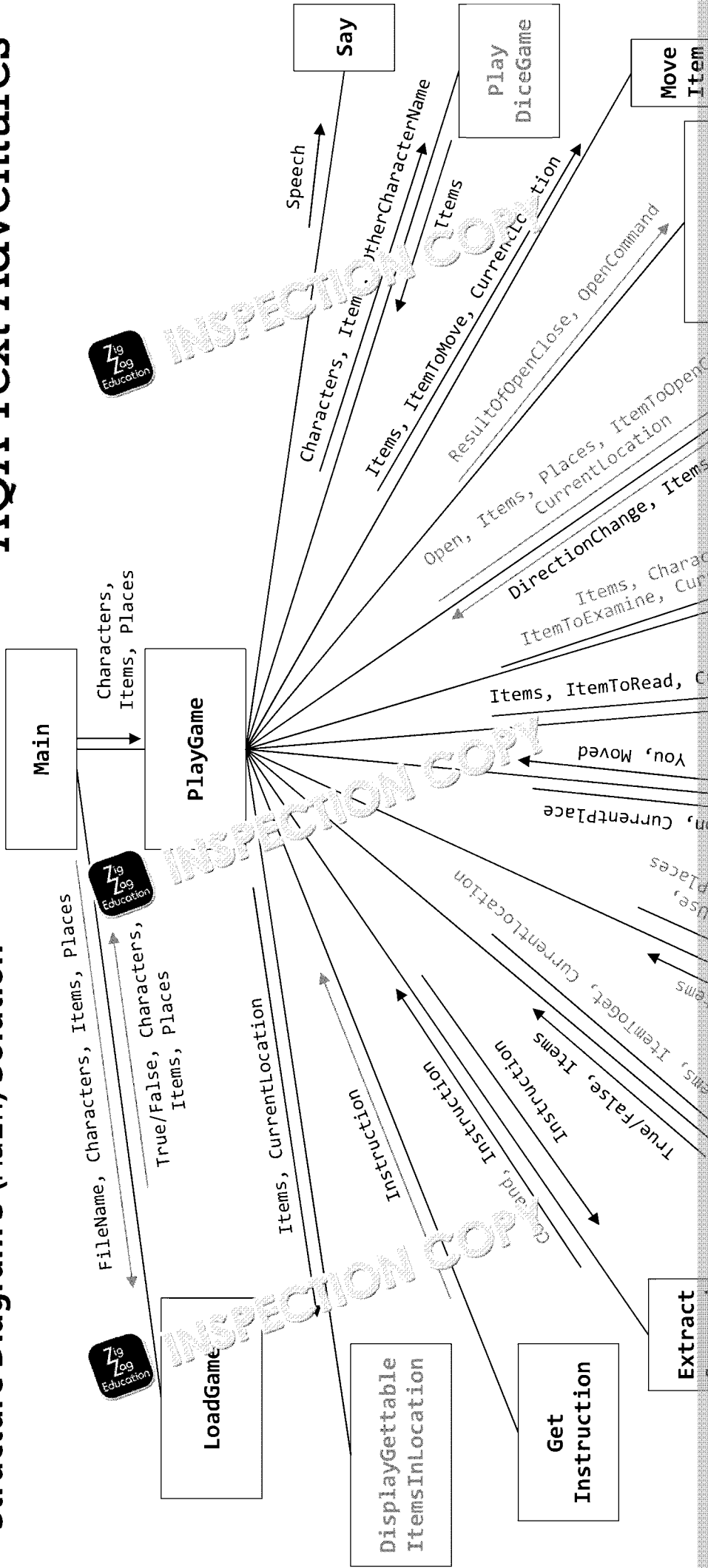


INSPECTION COPY

COPYRIGHT  
PROTECTED







COPYRIGHT  
PROTECTED



INSPECTION COPY

# AQA Text Adventures

## Written Questions (Python) - Solutions

| Q  | Answer/Markance                                                                                                                                                                                                                                                                                   |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1a | 1 mark: <ul style="list-style-type: none"> <li>• LowerLimitValue</li> <li>• UpperLimitValue</li> </ul>                                                                                                                                                                                            |
| 1b | 1 mark: <ul style="list-style-type: none"> <li>• ExtractCommand</li> <li>• ExtractCommandForCommand</li> <li>• OpenClose</li> <li>• UseItem</li> <li>• GetItem</li> </ul>                                                                                                                         |
| 1c | 4 marks: <ul style="list-style-type: none"> <li>• sys.stdout.write</li> <li>• len</li> <li>• raw_input</li> <li>• index</li> <li>• append</li> </ul>                                                                                                                                              |
| 1d | Character                                                                                                                                                                                                                                                                                         |
| 1e | 1 mark: <ul style="list-style-type: none"> <li>• Place</li> <li>• Character</li> <li>• Item</li> </ul>                                                                                                                                                                                            |
| 1f | 1 mark: <ul style="list-style-type: none"> <li>• ITEM_ID</li> <li>• ITEM_ID_FOR_ITEM</li> <li>• ID_DIFFERENCE_FOR_OBJECT_IN_TWO_LOCATIONS</li> </ul>                                                                                                                                              |
| 1g | 2 marks: <ul style="list-style-type: none"> <li>• ChangeStatusOfDoor</li> <li>• UseItem</li> <li>• LoadGame</li> </ul> Reject Examine, as it is a procedure and not a function.                                                                                                                   |
| 1h | GetInstruction                                                                                                                                                                                                                                                                                    |
| 2  | 2 marks: <ul style="list-style-type: none"> <li>• Converts the user input to lowercase</li> <li>• Because subsequent comparisons will be made with lowercase strings / to from being compared with otherwise identical lowercase strings</li> </ul>                                               |
| 3  | 3 marks: <ul style="list-style-type: none"> <li>• Stores whether the player has successfully moved</li> <li>• If a player attempts a move, it is set to <code>False</code></li> <li>• If it remains <code>False</code>, the message (you are not able to go in that direction) appears</li> </ul> |
| 4  | 2 marks: <ul style="list-style-type: none"> <li>• Control goes to the while loop</li> <li>• ...To return the current value of <code>ResultForCommand</code></li> </ul>                                                                                                                            |

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



| Q                  | Answer/Guidance                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 5                  | 2 marks: <ul style="list-style-type: none"> <li>• (Process of) connecting/joining two (or more) strings together</li> <li>• Any instance of a + used between two strings, e.g.               <ul style="list-style-type: none"> <li>◦ <code>ListOfItems += ", ";</code></li> <li>◦ <code>"You can't move " + ItemToMove + "."</code> (Does not)</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 6                  | 5 marks: <ul style="list-style-type: none"> <li>• Prints empty string</li> <li>• Prints 'You are currently carrying the following items:'</li> <li>• Loops through all items if the same</li> <li>• If an item is in the INVENTORY...</li> <li>• Print the name of that item</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 7                  | 3 marks: <ul style="list-style-type: none"> <li>• Temporarily holds an instance of <code>Character</code></li> <li>• Attributes are set by reading data from the file</li> <li>• Once attributes are set it is added to the variable <code>Characters</code></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 8                  | 3 marks: <ul style="list-style-type: none"> <li>• It may or may not be set to a different value in the following <code>if</code> clause</li> <li>• If it is not set to a value, then the following <code>while</code> loop would cause an error as <code>LowerLimitValue</code> was checked</li> <li>• Initialising the value to 0 prevents this error from occurring</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 9                  | 3 marks (first <code>while</code> loop): <ul style="list-style-type: none"> <li>• The first <code>while</code> loop checks for there being no space in the first character</li> <li>• If there is not a space, the first character is added to <code>command</code></li> <li>• The first character is (also) discarded from <code>Instruction</code> / the <code>Instruction</code> becomes a substring from the second character onward</li> <li>• Once the loop is complete, the <code>Instruction</code> variable contains everything from the second character onward and the <code>command</code> variable contains the contents of the initial string before the first space</li> </ul> 2 marks (second <code>while</code> loop): <ul style="list-style-type: none"> <li>• The second <code>while</code> loop checks for a space (in the first character of what is left of <code>Instruction</code>)</li> <li>• If there is a space, it is discarded</li> <li>• This loop removes leading spaces from <code>Instruction</code></li> </ul> |
| 10                 | 4 marks: <ul style="list-style-type: none"> <li>• Loops through each item in the game</li> <li>• If an item is in <code>INVENTORY</code> and has 'die' in its name...</li> <li>• ...Sets <code>PlayerHasDie</code> to <code>True</code></li> <li>• ...And sets <code>IndexOfPlayerDie</code> to the index of the item</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 11                 | 4 marks: <ul style="list-style-type: none"> <li>• Reference made to changing procedure <code>DisplayInventory</code></li> <li>• Selection statement positioned before 'You are currently carrying the following items:'</li> <li>• Selection statement checks if any items have the location of <code>INVENTORY</code></li> <li>• If none of the items are in <code>INVENTORY</code>, display alternative message, otherwise display the items in the list</li> </ul> <p>Alternative implementations potentially worth full marks would include the selection statement positioned after the message, or checking for items in the list in the opposite order (i.e. checking that it is not empty before the selection statement instruction).</p>                                                                                                                                                                                                                                                                                               |
| 12                 | 3 marks: <ul style="list-style-type: none"> <li>• Designing a <code>while</code> statement for instruction variable to be placed within a loop</li> <li>• A check to be made within the loop that the length of the string is at least 2 characters</li> <li>• A <code>continue</code> statement to continue until that check returns <code>True</code> / length is at least 2 characters</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 13                 | 2 marks: <ul style="list-style-type: none"> <li>• Execution would move to the <code>except</code> block / an exception would be thrown</li> <li>• The function <code>LoadGame</code> would return a value of <code>False</code> along with <code>Character</code></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>TOTAL MARKS</b> |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

COPYRIGHT  
PROTECTED

# AQA Text Adventures

## Programming Tasks (Python) - Suggested Solutions and Mark Scheme

Note that the following are recommended solutions and not an exhaustive list of solutions for each task. The marking guidance should be used as a guide only. Discretion should be used if alternative solutions are given.

### Task 1



**1 mark** Correct declaration of ShowHelp, with no parameters

**1 mark** Valid call to print, sys.stdout.write or Say with text indicated in the question. The wording **must match**, including the line break, although any alternative that uses two lines is acceptable.

```
def ShowHelp():  
    print "You can enter one of the following commands:"  
    print "get, use, examine, say, quit, read, move, open, close"
```

**1 mark** Additional elif clause in an appropriate location within PlayGame

```
elif Command == "help":  
    ShowHelp()
```

**1 mark** Screenshot showing output that matches text added to ShowHelp

```
Enter name> flag1  
  
You are in a room with blue walls. There is a green door  
in the eastern wall. There is a Persian rug on the floor.  
On the floor there is: torch, red die.  
  
> help  
You can enter one of the following commands:  
get, use, examine, say, quit, read, move, open, close  
>
```



**COPYRIGHT  
PROTECTED**



## Task 2

1 mark Additional argument in call to Examine from PlayGame

```
elif Command == "examine":  
    Examine(Items, Characters, Places, Instruction, Chara
```

1 mark Additional parameter added to Examine's definition

1 mark Addition to selection structure (5112) to accommodate additional edge case (A. 'room' code is added in if clause, elif clause or else clause)

1 mark Current location's description displayed in appropriate part of selection structure

1 mark Valid call to DisplayGettableItemsInLocation in appropriate part of selection structure

1 mark Examining inventory and other objects remains unaffected

```
def Examine(Items, Characters, Places, ItemToExamine, CurrentLocation):  
    Count = 0  
    if ItemToExamine == "inventory":  
        DisplayInventory(Items)  
    elif ItemToExamine == "room":  
        print  
        print  
        print Places[Characters[0].CurrentLocation - 1].Description  
        DisplayGettableItemsInLocation(Items, Characters[0].CurrentLocation)  
    else:  
        IndexOfItem = GetIndexOfItem(ItemToExamine, -1, Items)
```

1 mark Screenshot showing that room description after user enters 'examine room'

```
Enter your command: flag1  
  
You are in a room with blue walls. There is a green door in the north wall.  
There is a green door in the eastern wall. There is a Persian rug on the floor.  
On the floor there is: torch, red die.  
  
> examine room  
  
You are in a room with blue walls. There is a green door in the north wall.  
There is a green door in the eastern wall. There is a Persian rug on the floor.  
On the floor there is: torch, red die.  
  
>
```

INSPECTION COPY

COPYRIGHT  
PROTECTED



### Task 3

- 1 mark** Code within "quit" elif clause to display prompt 'Are you sure? (y/n)' (1)
- 1 mark** User input is stored in an appropriately named variable
- 1 mark** Selection clause that catches either 'y' or 'yes'
- 1 mark** Selection clause catches both 'y' and 'yes' in a combination of cases, either lower (either done automatically by calling GetInstruction() or manually raw\_input()) or by checking for every possible combination
- 1 mark** Output and Boolean assignment inside the selection structure and break

```
elif Command == "quit":  
    Say("Are you sure? (y/n)")  
    response = GetInstruction()  
    if response == "y" or response == "yes":  
        Say("You decide to give up, try again another time")  
        StopGame = True
```

- 1 mark** Output showing 'no' resulting in a prompt, followed by 'yes' resulting in terminating after the output

```
Enter filename> flag1  
  
You are in a room with blue walls. There is a green  
door in the eastern wall. There is a Persian rug  
On the floor there is: a sword and a die.  
  
> quit  
Are you sure? (y/n)  
  
> no  
  
> quit  
Are you sure? (y/n)  
  
> yes  
You decide to give up, try again another time
```

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



## Task 4

- 1 mark** Attempt to convert user input to an integer within the LowerLimitValue try except statement (A. if done for either LowerLimitValue or UpperLimitValue)
- 1 mark** LowerLimitValue set to user input if an integer was entered (A. if done for either LowerLimitValue or UpperLimitValue)
- 1 mark** Loop should continue if a non-integer value entered (A. if done for either LowerLimitValue or UpperLimitValue)
- 1 mark** For a value above for both LowerLimitValue and UpperLimitValue

```
def playdice(Lower, Upper):
    LowerLimitValue = 0
    if Lower.isnumeric():
        LowerLimitValue = int(Lower)
    else:
        while LowerLimitValue < 1 or LowerLimitValue > 6:
            try:
                LowerLimitValue = int(raw_input("Enter minimum: "))
            except:
                LowerLimitValue = 0
    UpperLimitValue = 0
    if Upper.isnumeric():
        UpperLimitValue = int(Upper)
    else:
        while UpperLimitValue < LowerLimitValue or UpperLimitValue > 6:
            try:
                UpperLimitValue = int(raw_input("Enter maximum: "))
            except:
                UpperLimitValue = 0
    G = randomNumber(LowerLimitValue, UpperLimitValue)
```

- 1 mark** Shows 'one' and '1.0' not being accepted for the minimum value, and '1' being accepted for the minimum
- 1 mark** Shows 'six' and '6.0' not being accepted for the maximum value, and '6' being accepted for the maximum

```
Enter filename> flag2

Welcome

You are in a guardroom. There is a door leading to a jail cell in the north. There is a corridor going eastward.

> playdice guard
Enter minimum: one
Enter minimum: 1.0
Enter minimum: 1
Enter minimum: 1.0
Enter minimum: 6
You rolled a 3.
They rolled a 3.
Draw!

>
```

INSPECTION COPY

COPYRIGHT  
PROTECTED



## Task 5

1 mark Declaring temporary list of inventory items

1 mark Loop to add all inventory items to the temporary list

```
def DisplayInventory(Items):  
    InventoryItems = []  
    for Thing in Items:  
        if Thing.Location == INVENTORY:  
            InventoryItems.append(Thing)
```

1 mark Loop to iterate through each pass of the sort operation (x in this example)

1 mark Loop to iterate through each pair for individual comparisons (y in this example)

1 mark Nested iteration would cause a comparison to be made beyond the list's length

1 mark Selection structure that attempts to compare adjacent values, even if syntax is wrong

1 mark Selection structure accesses the Name attribute of an item for the comparison

1 mark Selection structure is valid, comparing appropriate name values and adding to temporary list

1 mark Temporary variable used to store one of the items to be switched inside selection structure

1 mark Pair of items switched inside selection structure

```
for x in range(0, len(InventoryItems) - 1):  
    for y in range(0, len(InventoryItems) - 1):  
        if InventoryItems[y].Name.lower() > InventoryItems[y+1].Name.lower():  
            temp = InventoryItems[y]  
            InventoryItems[y] = InventoryItems[y+1]  
            InventoryItems[y+1] = temp
```

1 mark Displaying items in the order they are output in the sorted array

```
print  
print "You are currently carrying the following items:"  
for Thing in InventoryItems:  
    if Thing.Location == INVENTORY:  
        print Thing.Name
```

1 mark Screenshot showing flag1 being loaded, 'examine inventory' being entered, (apple, flask, jar) being displayed in alphabetical order. After 'get red die' and 'examine inventory' the inventory should again be displayed in alphabetical order (apple, flask, jar, red die)

```
Enter filename> flag1  
  
You are in a room with blue walls. There is a green door in the eastern wall. There is a Persian rug on the floor there is: torch, red die.  
  
> examine inventory  
  
You are currently carrying the following items:  
apple  
flask  
jar  
  
> get red die  
You have got the red die.  
  
> examine inventory  
  
You are currently carrying the following items:  
apple  
flask  
jar  
red die
```

INSPECTION COPY

COPYRIGHT  
PROTECTED



## Task 6

1 mark Constant correctly declared and set to 4

```
INVENTORY = 1001
MINIMUM_ID_FOR_ITEM = 2001
ID_DIFFERENCE_FOR_OBJECT_IN_TWO_LOCATIONS = 10000
MAX_INVENTORY = 4
```

1 mark Variable to count inventory items

1 mark Iteration through items

1 mark Selection statement or equivalent to identify items in the inventory

1 mark Incrementing variable inside selection statement

```
inventorySize = 0
for Thing in Items:
    if Thing.Location == INVENTORY:
        inventorySize += 1
```

1 mark Selection statement to check that inventorySize >= constant (R. ==)

1 mark 'Inventory full' displayed only under correct conditions when the player tries to take an item

1 mark Remainder of selection structure would not be evaluated if inventory is full (or equivalent)

```
if inventorySize >= MAX_INVENTORY:
    print "Inventory full"
    CanGet = False
```

1 mark Implementing inventorySize check in both GetItem and TakeItem

1 mark 'Inventory full' displayed only under correct conditions when the player tries to take an item

```
inventorySize = 0
for Thing in Items:
    if Thing.Location == INVENTORY:
        inventorySize += 1
if inventorySize >= MAX_INVENTORY:
    print "Inventory full"
elif ChosenItem in ListOfNamesOfItemsInInventory:
    print "You have that now."
    Pos = ListOfNamesOfItemsInInventory.index(ChosenItem)
    Items = ChangeLocationOfItem(Items, ListOfIndicesOfItemsInInventory, Pos)
else:
    print "They don't have that item, so you don't take anything"
```

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



- 1 mark** First screenshot shows torch being picked up, red die not being picked up accordingly

```
Enter filename> flag1

You are in a room with blue walls. There is a green door in the eastern wall. There is a Persian rug on the floor. On the floor there is: torch, red die.

> get torch
You have that now.

> get red die
Inventory full

> examine inventory

You are currently carrying the following items:
flask
apple
jar
torch
```

- 1 mark** Second screenshot shows dice game being played, jar won, jar not being picked up accordingly

```
Enter filename> flag1

You are in a guardroom. There is a metal silver door leading to a jail cell in the northern wall. There is a corridor going eastwards.

> playdice guard
Enter minimum: 6
Enter maximum: 6
You rolled a 6.
They rolled a 5.
You win!
Which item do you want to take? They have: jar, gold key
jar
Inventory full

>
```

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



# SPECTACULAR COPY



## Task 8

- 1 mark Variable to track whether loop should repeat, or equivalent
- 1 mark Loop repeats until a file name has successfully loaded
- 1 mark Prompt amended to include quit option
- 1 mark Selection (or inclusion in a loop) to address entry of 'Q'
- 1 mark Execution always terminates when 'Q' is entered
- 1 mark Selection statement to determine if '.gme' within file name
- 1 mark '.gme' added only if it is not already within the string
- 1 mark Successfully loading a file ensures loop would not run again

```
GameLoaded = False
while not GameLoaded:
    Filename = raw_input("Enter filename or enter Q to quit> ")
    if Filename == "Q":
        break
    if ".gme" not in Filename:
        Filename += ".gme"
    print
    GameLoaded, Characters, Items, Places = LoadGame(Filename,
    if GameLoaded:
        PlayGame(Characters, Items, Places)
    else:
        print "Unable to load game."
```

- 1 mark 'flag3' causes prompt to repeat (i.e. extra line results)
- 1 mark 'flag2.gme' causes game to begin

```
Enter filename or enter Q to quit> flag3
Unable to load game.
Enter filename or enter Q to quit> flag2.gme

You are in a guardroom. There is a metal silver
leading to a jail cell in the northern wall. T
. There is a corridor going eastwards.

>
```

INSPECTION COPY

COPYRIGHT  
PROTECTED



## Task 9

- 1 mark DisplayCharacters correctly declared with one parameter
- 1 mark Header row contains names of all fields as appropriate
- 1 mark Spacing between headers is correct
- 1 mark Loop to iterate through each element of Characters
- 1 mark Each entry (even if inaccurate) will display on a separate line
- 1 mark All four attributes displayed on one line, with no additional text
- 1 mark Spacing will be correct regardless of size of a field

```
def DisplayCharacters(Characters):  
    Header = ""  
    Header += "%-5s" % ("ID",)  
    Header += "%-12s" % ("NAME",)  
    Header += "%-64s" % ("DESCRIPTION",)  
    Header += "CURRENTLOCATION"  
    print Header  
    for Character in Characters:  
        Line = ""  
        Line += "%-5s" % Character.ID  
        Line += "%-12s" % Character.Name  
        Line += "%-64s" % Character.Description  
        Line += "%d" % Character.CurrentLocation  
        print Line
```

- 1 mark Selection clause adapted to include checking for input of 'debugchar'
- 1 mark Valid call to DisplayCharacters from PlayGame

```
elif Command == "debugchar":  
    DisplayCharacters(Characters)
```

- 1 mark Two characters displayed, lined up correctly in columns in the correct order

```
Enter filename> flag1  
  
You are in a room with blue walls. There is a green door in the north  
door in the eastern wall. There is a Persian rug on the floor.  
On the floor there is: torch, red die.  
  
> debugchar  
ID      NAME      DESCRIPTION  
1001 me      You think you look OK  
1002 guard    The guard is about 180cm tall. He is wearing a red  
>
```

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



INSPECTION COPY



## Task 10

- 1 mark `FillVessel` correctly declared with four parameters
- 1 mark Variable, or equivalent, to store whether the item is a container
- 1 mark Variable, or equivalent, to store whether the item is large (*equivalence in 'IsContainer' is set to true only if the item is both a container and not conditions to share a single variable, and all parts of the code above 2 marks can still be awarded*)
- 1 mark Variable, or equivalent, to store whether the item is in the player's inventory
- 1 mark Variable, or equivalent, to store whether the player is in the same location
- 1 mark Loop, or equivalent, to process items in turn (A. multiple loops, calls to `GetIndexOfType`)
- 1 mark Selection statement to check whether the vessel is a container
- 1 mark Selection statement to check whether the vessel is (not) large
- 1 mark Selection statement to check whether the vessel is in the player's inventory
- 1 mark Selection statement to check whether the player and the barrel are in the same location
- 1 mark All variables are set to the correct values under all circumstances

```
def FillVessel(Characters, Items, Places, Vessel):
    IsContainer = False
    InInventory = False
    ByBarrel = False
    Fillable = None
    for Thing in Items:
        if Vessel == Thing.Name and "container" in Thing.Status and
            IsContainer = True
            Fillable = Thing
        if Vessel == Thing.Name and Thing.Location == INVENTORY:
            InInventory = True
        if Thing.Name == "Barrel" and Thing.Location == Characters[0].Location:
            ByBarrel = True
```

- 1 mark Selection statement to check that all conditions are met
- 1 mark Inside selection statement, append ' of water' to the vessel's name (A. append the item or call to `GetIndexOfType`; this example code uses a variable defined above)
- 1 mark 'You have filled the ' with name of vessel appended is displayed if the vessel is filled
- 1 mark 'You cannot do that' displayed if the vessel has not been filled
- 1 mark Check for presence of 'of water' within the name attribute
- 1 mark For ensuring that the program will not crash even if the `Fillable` is not set
- 1 mark 'Already full' displayed under the correct circumstances

```
if IsContainer and InInventory and ByBarrel and " of water" not in Fillable.Name:
    Fillable.Name = Vessel + " of water"
    print "You have filled " + Vessel + " with water"
elif Fillable is None and " of water" in Fillable.Name:
    print "Already full"
else:
    print "You cannot do that"
```

- 1 mark Selection structure in `PlayGame` adjusted to include 'fill' command

**COPYRIGHT  
PROTECTED**



**1 mark** Valid call to `FillVessel` at valid point in selection structure

```
elif Command == "fill":  
    FillVessel(Characters, Items, Places, Instruction)
```

**1 mark** Input of 'fill flask' displays 'You have filled the flask' and 'fill apple' displays

**1 mark** Input of 'fill flask of water' displays 'Already full' and 'examine inventory'

```
You are in a cellar. There is a large jar sitting up. The jar is empty.  
  
> fill flask  
You have filled the flask.  
  
> fill apple  
You cannot do that.  
  
> fill flask of water  
Already full.  
  
> examine inventory  
You are currently carrying the following items:  
flask of water  
apple  
jar  
  
>
```

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



## Task 11

1 mark History class correctly declared

1 mark Integer variable Pointer and integer array Locations correctly declared

1 mark Pointer and Locations initialised to -1 and 5 elements respectively  
be declared as list of 5 None elements as long as Push doesn't allow for more than 5 elements, and Locations hasn't been initialised with any other values

```
class History():  
    def __init__(self):  
        self.Pointer = -1  
        self.Locations = [None] * 5
```

1 mark Push correctly declared within the History class with two parameters

1 mark Selection statement to check whether Pointer is < 4 or <= 3

1 mark If so, Pointer is incremented...

1 mark ...and the parameter is correctly inserted into the array

1 mark If not, contents of elements 4, 3, 2 and 1 are shifted down, with contents of element 0 being lost

1 mark ...and the parameter is inserted into element 4 of the array

```
def Push(self, NewLocation):  
    if self.Pointer < 4:  
        self.Pointer += 1  
        self.Locations[self.Pointer] = NewLocation  
    else:  
        for x in range(0, 3):  
            self.Locations[x] = self.Locations[x+1]  
        self.Locations[4] = NewLocation
```

1 mark Pop correctly declared within the History class with one parameter

1 mark Check for a Pointer value of -1

1 mark If the Pointer is set to -1, then -1 should be the return value

1 mark Pointer is decremented

1 mark Value that was at the index previously indicated by Pointer is returned

```
def Pop(self):  
    if self.Pointer > -1:  
        self.Pointer -= 1  
        return self.Locations[self.Pointer + 1]  
    else:  
        return -1
```

1 mark Past declared as an instance of History in PlayGame

```
def PlayGame(Characters, Moves, Places):  
    Past = History()  
    StopGame = False  
    while not StopGame:
```

1 mark Past instance passed to Go when 'go' command entered by user (A. any other command will not be processed)

```
        elif Command == "go":  
            Characters[0], Moved = Go(Characters[0], Instruction, Places[Characters[0].CurrentLocation - 1], Past)
```

INSPECTION COPY

COPYRIGHT  
PROTECTED



**1 mark** Declaration of Go now includes a parameter of type History in same p

```
def Go(You, Direction, CurrentPlace, Past):
```

**1 mark** Attempt to call Push within at least one if clause (even if unsuccessful)

**1 mark** Push called correctly in all six cases, before player is moved to the new location, then variable pushed after movement should contain all six calls, each of which will be structured as below:

```
if Direction == "north":
    if CurrentPlace != 0:
        Moved = 1
        You.Push(You.CurrentLocation)
        You.CurrentLocation = CurrentPlace.North
```

**1 mark** New if clause added for 'back'

**1 mark** Call to Pop (R. if any pass could call Pop repeatedly)

**1 mark** Selection statement to address return value of -1

**1 mark** Message 'You cannot go back' displayed if -1 has been returned

**1 mark** Location set to popped value only if it was not -1

```
elif Direction == "back":
    NewLocation = Past.Pop()
    if NewLocation != -1:
        You.CurrentLocation = NewLocation
    else:
        print "You cannot go back"
```

**1 mark** Having gone East to flag2, the original room description should re-display second time. If result in 'You cannot go back' being displayed.

```
Enter name> flag2

You are in a guardroom. There is a metal silver door leading to a jail cell
leading to a jail cell in the northern wall. There is a clear desk with a
. There is a corridor going eastwards.

> go east

You are in a long, empty corridor. There is a yellow door in the eastern wall.

> go back

You are in a guardroom. There is a metal silver door leading to a jail cell
leading to a jail cell in the northern wall. There is a clear desk with a
. There is a corridor going eastwards.

> go back
You cannot go back.

You are in a guardroom. There is a metal silver door leading to a jail cell
leading to a jail cell in the northern wall. There is a clear desk with a
. There is a corridor going eastwards.

>
```

**COPYRIGHT  
PROTECTED**



## Task 12

**1 mark** Addition of Health attribute to Character class

**1 mark** Characters given Health of 10 by default

```
class Character():
    def __init__(self):
        self.Name = self.Description = ""
        self.ID = self.CurrentLocation = 0
        self.Health = 10
```

**1 mark** EatItem correctly declared with three parameters

**1 mark** Index of item acquired by iteration or call to GetIndexOfItem

**1 mark** Index of -1 would not throw an exception

**1 mark** Selection checks for 'edible' within the status of an item (A. if wrong item)

**1 mark** Selection checks that an item is in the player's inventory (A. if wrong item)

**1 mark** Five added to Health only if item is edible and in the inventory (R. if wrong item)

**1 mark** Output indicates that the item was eaten and new Health value displayed

**1 mark** 'You cannot eat that' displayed in all other circumstances, including if an non-existent or misspelled item) was 'eaten'.

```
def EatItem(Characters, Items, ItemToEat):
    ItemIndex = GetIndexOfItem(ItemToEat, -1, Items)
    if ItemIndex > -1:
        if "edible" in Items[ItemIndex].Status and Items[ItemIndex].InInventory(Characters):
            Characters[0].Health += 5
            del Items[ItemIndex]
            print "You have eaten the " + Items[ItemIndex].Name
            print "Your health is %d" % Characters[0].Health
        else:
            print "You cannot eat that"
    else:
        print "You cannot eat that"
```

**1 mark** Health displayed after call to DisplayGettableItemsInLocation

```
DisplayGettableItemsInLocation(Items, Characters[0].CurrentLocation)
print "Your health is %d" % Characters[0].Health
Moved = False
```

**1 mark** Valid call to EatItem within PlayGame

```
elif Command == "eat":
    EatItem(Characters, Items, Instruction)
```

**1 mark** Health initially output as 10; first response to 'eat apple' is that the apple is now 15; second call to 'eat apple' results in 'You cannot eat that'

```
Enter filename> flag1

You are in a room with blue walls. There is a door in the east. There is a Persian rug on the floor. There is a red die.
You are holding nothing.

> eat apple
You have eaten the apple
Your health is 15

> eat apple
You cannot eat that
```

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



## Task 13

- 1 mark Variable to track whether item has been taken (A. appropriate break or
- 1 mark Loop to begin before Count = 1;
- 1 mark An item being successfully taken will terminate the loop
- 1 mark An item not being taken will cause the loop to re-run
- 1 mark Output 'They don't have a...' correct including concatenation with ChosenItem

```
ItemTaken = False
while not ItemTaken:
    sys.stdout.write("Which item do you want to take? They have: ")
    sys.stdout.write(ListOfNamesOfItemsInInventory[0])
    while Count < len(ListOfNamesOfItemsInInventory) - 1:
        sys.stdout.write(", ", ListOfNamesOfItemsInInventory[Count])
        Count += 1
    print ", "
    ChosenItem = input()
    if ChosenItem in ListOfNamesOfItemsInInventory:
        ItemTaken = True
        print "You have that now."
        Pos = ListOfNamesOfItemsInInventory.index(ChosenItem)
        Items = ChangeLocationOfItem(Items, ListOfIndicesOfItemsInInventory, Pos)
    else:
        print "They don't have a %s" % ChosenItem + " - try again"
```

- 1 mark Attempt to take a box answered with 'they don't have a box - try again'; with 'You have that now'.

```
Enter filename> fl_g2
You are in a guardroom. There is a metal silver door
leading to a jail cell in the northern wall. There is
. There is a corridor going eastwards.

> playdice guard
Enter minimum: 6
Enter maximum: 6
You rolled a 6.
They rolled a 3.
You win!
Which item do you want to take? They have:jar, gold k
box
They don't have a box - try again.
Which item do you want to take? They have:jar, gold k
jar
You have that now.
```

INSPECTION COPY

COPYRIGHT  
PROTECTED



## Task 14

- 1 mark** Selection structure within `CheckIfDiceGamePossible` after `PlayerHasDie`, `PlayersInSameRoom` and `OtherCharacterHasDie` have been assigned as `boolean` statements
- 1 mark** 'Player has no die' displayed only if `PlayerHasDie` is *false*
- 1 mark** '\_\_\_\_ is not here' displayed only if `PlayersInSameRoom` is *false*, to indicate that the player is not in the same room as the other character
- 1 mark** '\_\_\_\_ has no die' displayed only if `OtherCharacterHasDie` is *false* (the `else` from the previous mark was denied due to a failure in concatenation)

```
if not PlayerHasDie:
    print "Player has no die"
elif not PlayersInSameRoom:
    print OtherCharacterName + " is not here"
elif not OtherCharacterHasDie:
    print OtherCharacterName + " has no die"
return PlayerHasDie and PlayersInSameRoom and OtherCharacterHasDie
IndexOfPlayerDie, IndexOfOtherCharacter, IndexOfOtherCharacterDie
```

- 1 mark** 'Player has no die' on first attempt; 'guard is not here' on second attempt; 'You can't play a dice game.' should still appear both times (R. if any additional output)

```
Enter filename> flag1

You are in a room with blue walls. There is a door in the eastern wall. There is a Persian rug on the floor. On the floor there is: torch, red die.

> playdice guard
Player has no die
You can't play a dice game.

> get die
You have got that now.

> playdice guard
guard is not here
You can't play a dice game.

>
```

**COPYRIGHT  
PROTECTED**



## Task 15

**1 mark** Item names are padded with space to make them the same width

**1 mark** Descriptions are displayed on the same line as names (R. if multiple inven

**1 mark** Descriptions, but not names, are displayed in round brackets

```
def DisplayInventory(Items):
    print
    print "You are currently carrying the following items:"
    for Thing in Items:
        if Thing.Inventory < INVENTORY:
            print "%-10s" % (Thing.Name,)
            Line += " "
            Line += Thing.Description
            Line += "\n"
    print Line
```

**1 mark** Descriptions in brackets in a straight-line column

```
Enter filename> flag1

You are in a room with blue walls.  There is a green door in the
door in the eastern wall.  There is a Persian rug on the floor.
On the floor there is: torch, red die.

> examine inventory

You are currently carrying the following items:
flask      (It is an empty plastic flask.)
apple      (It is an apple.  It looks tasty.)
jar        (It is an empty glass jar.  The lid is missing.)

>
```

INSPECTION COPY

**COPYRIGHT  
PROTECTED**

INSPECTION COPY



Name

ZigZag Education supporting

# A Level AQA Computer Science Paper

Summer 2019

## AQA Text Adventures

### Electronic Answer Document (EAD)

#### Instructions

- Enter your name in the box at the top of this page
- Answer **all** questions by entering your answers into this document
- Remember to **save** this document regularly
- Save and print this document and any additional pages
- Answer **all** questions
- The marks available for each question are shown in brackets
- You will need:
  - ☐ access to a computer
  - ☐ access to a printer
  - ☐ access to appropriate software
  - ☐ electronic copies of the required skeleton code
  - ☐ EAD (Electronic Answer Document)

Total marks:

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



## Written Questions

Answer all questions.  
Remember to save this document regularly.

| Q  |     |
|----|-----|
| 1  | (a) |
|    | (b) |
|    | (c) |
|    | (d) |
|    | (e) |
|    | (f) |
|    | (g) |
|    | (h) |
| 2  |     |
| 3  |     |
| 4  |     |
| 5  |     |
| 6  |     |
| 7  |     |
| 8  |     |
| 9  |     |
| 10 |     |
| 11 |     |
| 12 |     |
| 13 |     |

INSPECTION COPY

**COPYRIGHT  
PROTECTED**



## Programming Tasks

Answer all questions.

Remember to save this document regularly.

| Q  | Answer |
|----|--------|
| 1  |        |
| 2  |        |
| 3  |        |
| 4  |        |
| 5  |        |
| 6  |        |
| 7  |        |
| 8  |        |
| 9  |        |
| 10 |        |
| 11 |        |
| 12 |        |
| 13 |        |
| 14 |        |
| 15 |        |

INSPECTION COPY

**COPYRIGHT  
PROTECTED**

