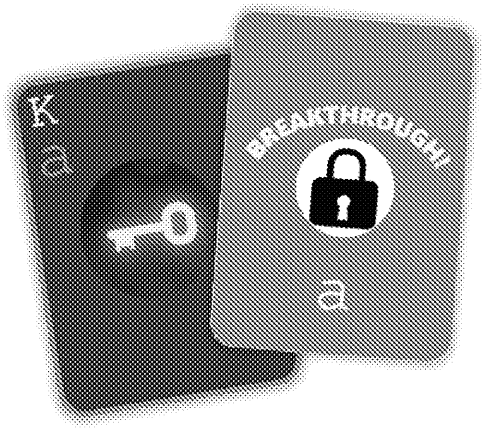
Zig Zag Education

Computer Science

A Level | AQA | 7517

2015 specification
for the 2022 exam

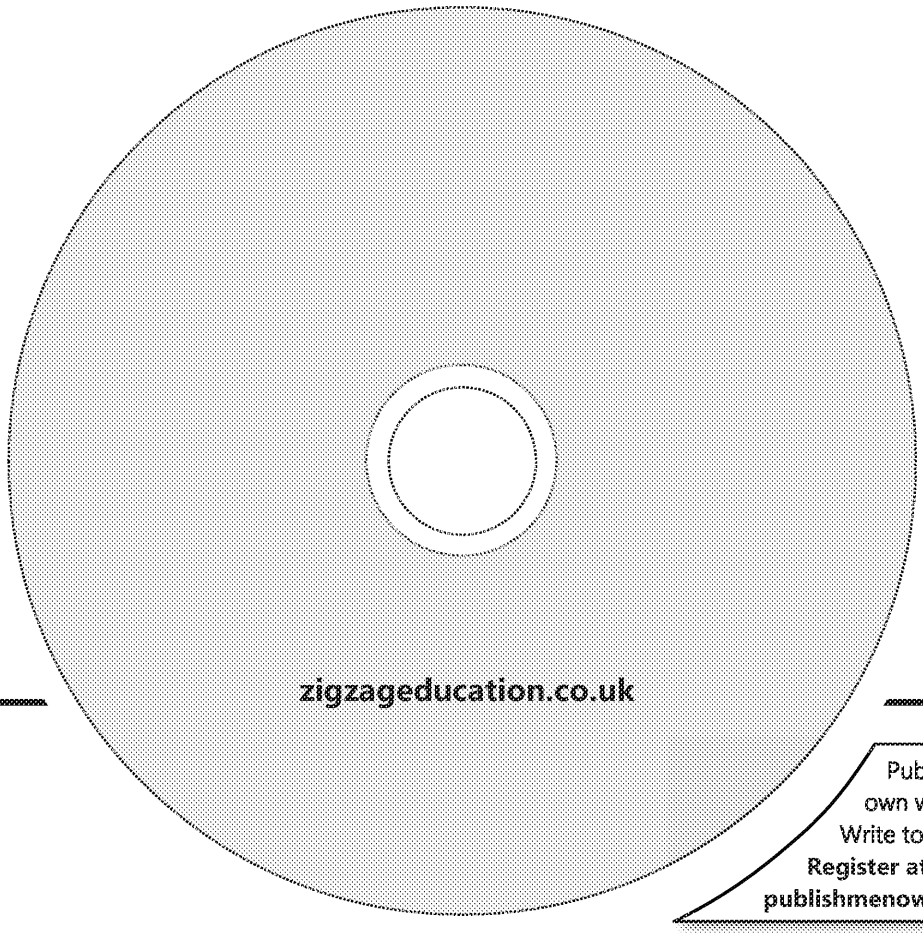


PAPER 1 EXAM RESOURCE PACK 2022

BREAKTHROUGH!

for A Level AQA Computer Science

VB.NET EDITION



DH9/ 11150

POD 11150

zigzageducation.co.uk

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

Contents

** Note there are also electronic copies of the UML Diagrams ('Complete' & 'Activity' versions) on the CD – which can be printed in A3, making them much more usable (especially when used as activities)*

Teacher's Introduction

This resource pack is designed to help you support your students taking the **A Level Computer Science Paper 1** exam. It is based on the ***Breakthrough!*** preliminary material (VB.NET) – for examination summer 2022.

On the CD, you will find the following:

`Breakthrough`	this folder contains all of the content (PDF/DOCX) accessible via a HTML interface
`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 `Breakthrough` 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 accessible via the *Solutions* web page are password-protected, so that students can only access them with your permission. Each password is a four-digit code, as follows:

The resource pack consists of the following:

① **Code Breakdown**
This document gives a detailed technical overview of the skeleton program, describing in detail each class and method in turn – including their purpose/function, parameters and return values.

> **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.

② **Class Diagrams**
Three UML Class Diagrams help students explore the skeleton program; there is a completed version, a partially-complete version (gap-fill), as well as a mostly blank template. The completed version is password-protected and accessible via the *Solutions* web page.

③ **Video**
Quick video going over the *Breakthrough!* card game mechanics – intended as a visual aid to accompany the notes in the official AQA preliminary material.

④ **Written Questions**
Theory questions testing students' understanding of the skeleton program. 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 versions are available. Suggested answers are provided via the *Solutions* web page as a password-protected PDF.

⑤ **Coding Tasks**
Fifteen modification exercises put students' programming skills to the test. Example solutions with suggested mark schemes are provided via the *Solutions* web page as a password-protected PDF. Note that these are example solutions and you must use your discretion to award marks accordingly where there are valid alternative solutions.

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. iii) before using it.**

BREAKTHROUGH

Skeleton Code Breakdow

Note: In the skeleton code released by AQA all parameters are

Class: Breakthrough

Identifier / Data		Description
<<constru		
Parameter	n/a	Initialises several private attributes including • Deck to a new CardCollection • Hand to a new CardCollection • Sequence to a new CardCollection • Discard to a new CardCollection • Score to 0 • GameOver to False • Locks to an empty list • CurrentLock to an empty Lock • LockSolved to False Invokes the LoadLocks() method to load the 'locks.txt'.
Return values	n/a	
AddDifficultyCardsToDeck (private)		
Parameters	n/a	Adds fi cyCards to the Deck.
Return values	n/a	
CheckIfL llengeMet (private)		
Parameters	n/a	Iterates through the Sequence CardCollectio string SequenceAsString with a comma and between each card description. As a new element from Sequence is concaten SequenceAsString, the string is compared w using the CheckIfConditionMet() method on whether a challenge has been met. This is tes challenges can be different lengths. If a challe returned, otherwise False is returned.
Return values	Boolean	
CheckIfPlayerHasLost (private)		
Parameters	n/a	Checks to see if the y cards left in the appropriate displayed on the scre the er and the method returns True ere are cards still left in the Deck, the play is returned, allowing the player to continue pla
Return values	Boolean	
CreateSta Deck (private)		
Parameters	n/a	Used by the SetupGame() method to populat correct File, Pick and Keys for each toolkit. 5 Picks from toolkits a, b and c are added to t 3 Keys from toolkits a, b and c are also added
Return values	n/a	

Identifier / Data		Description
GetCardChoice (private)		
Parameters	n/a	Used by the PlayGame() method t their Hand they would like to use.
Return values	Value : Integer	Contains error handling to catch n not catch a out of range.
GetCardFromDeck (private)		
Parameters	CardChoice : Integer	Used to get the next card from the add it to the Hand.
Return values	n	If the Deck CardCollection has a system will then check if the card a a DifficultyCard. If a DifficultyCa they would like to lose a 'Key' card cards from the Deck. The Difficul Discard CardCollection and the on the DifficultyCard passing the parameters.
		The system then performs a check repopulating the Hand with cards If another Difficulty card is found Difficulty card (or cards if there is the Deck) is move automatically i CardCollection rather than into th
		If the D an out of cards, the
GetChoice (private)		
Parameters	n/a	Used by the PlayGame() method like to use a card from their Hand CardCollection on the screen.
Return values	: String	
GetDiscardOrPlayChoice (private)		
Parameters	n/a	Used by the PlayGame() method like to play the selected card from Discard the selected card from the CardCollection.
Return values	Choice : String	
GetRandomLock (private)		
Parameters	n/a	Returns a randomly selected lock Locks.
Return values	Lock	
LoadGame (private)		
Parameters	FileName : String	Use the FileName parameter to imports the current Score, Challe for the Hand, Sequence, Discard
Return values	Boolean	True is returned if the file is loade error occurs, an error message is returned.

Identifier / Data		Description
LoadLocks (private)		
Parameters	n/a	Uses a hard-coded 'lo locks available for the file contains the challe lines from the file is spl Challenges, using a s Each Challenge is the as a delimiter into the The Conditions are th Lock variable – Locks to the private attribute If an error occurs, an e advise that the locks correctly.
Return values	n/a	
MoveCard (private)		
Parameters	FromCollection : CardCollection ToCollection : CardCollection CardNumber : Integer	Moves a card at the p the CardCollection F CardCollection ToC If the FromCollection ToCollection is the S been chosen (i.e. not score is updated appr ed. For all other m another, Score is not Score is returned.
Return values	Score : Integer	
PlayCardToSequence (private)		
Paramete	choice : Integer	This method is used t to the Sequence to te The system tests to s one card in the CardC system then checks t by the user is a differe played card. If the To can be played and the to the Sequence and appropriately for that then gets a new card Hand. If the Sequence does the system move Sequence and the Se The system then uses CheckIfLockChallen the new card added t Challenge to be met appropriate message the player Score by 5
Return va	n/a	

Identifier / Data		Description
PlayGame (public)		
Parameters	n/a	This contains the main game loop.
Return values	n/a	Checks to confirm if the private list attribute L loaded by the LoadLocks() method. If none displayed on the s a d the program qu If the lis s tain locks, it initialises the • LockSolved to False • Invokes the SetupGame() method to The main game loop runs while the private a There is then an inner loop which runs while private attribute LockSolved is also False. The inner game loop displays the current us current lock and the contents of the Hand, a CardCollections. Using the GetChoice() method to display a c game loop then uses selection to either disp CardCollection or use a card in the game. If the user selects to use a card, the system method to select a card. It then uses the Get method to confirm if the user wants to play o the user selects discard, the system moves t Hand to the Discard C Collection and ge using GetCard k m ck(). If the user selec PlayC Sequence() method to move th quence CardCollection. Once a card has been played or discarded, t GetLockSolved() method on the CurrentLo challenges have been met. If they have, the True and a new lock is generated. If a lock has been solved, the inner loop retu which checks if the game is over by invoking method. If this returns True the game ends.
ProcessLockSolved (private)		
Parameters	n/a	Increments the Score by 10 and displays the
Return values	n/a	Uses an indefinite loop to iterate through the returning all of the cards back to the Deck. Reshuffles the Deck using the Shuffle() met using the GetRando k() method with th CurrentLock.

Identifier / Data		Description
SetupCardCollectionFromGameFile (private)		
Parameters	LineFromFile : String CardCol : CardCollection	Used for processing lines 4 to file which are for processing t CardCollections (namely the Sequence).
Return values	n/a	
		Receives a single line of text parameter) from the external processes it into a CardColle LineFromFile contains text, i SplitLine, using the comma a The SplitLine list is then proc card number and card type in CardCollection. If a Difficult instead of a normal ToolCard
SetupGame (private)		
Parameters	n/a	Called from the PlayGame() message of the game on the would like to load in an extern game. If the player chooses to system attempts to load the fi be loaded the game quits. If the player chooses to play a generates a new Deck using method and then shuffles it by method. It then moves 5 cards start the player off. The system AddDifficultyCardsToDeck() DifficultyCards into the Deck ensure they are in random loc assigns a new lock at random CurrentLock using GetRand
Return values	n/a	
SetupLock (private)		
Parameters	Line1 : String Line2 : String	Used for processing lines 2 a file which contain the challeng The parameter Line1 contains and the parameter Line2 cont Each line is split into a string delimiter. The Line1 parameter is then the delimiter to add a new cha single line may contain multip parameter is split using a sem populate the Met status for e SetChallengesMet() method
Return values	n/a	

Class: Challenge

Identifier / Data		Description
<<constructor>>		
Parameters	n/a	Initialises the following p
Return values	n/a	Met to False • Conditions to an
GetCondition (public)		
Parameters	n/a	Returns a list of strings o challenge in the lock.
Return va	ondition : List (String)	
GetMet (public)		
Parameters	n/a	Returns the value of the p
Return values	Met : Boolean	
SetCondition (public)		
Parameters	NewCondition : List (String)	Sets the value of the pro Condition from the para
Return values	n/a	
SetMet (public)		
Parameters	NewValue : Boolean	Sets the value of the pro parameter NewValue.
Return values	n/a	

Class: Lock

This class does not have a specific constructor and the

Identifier /		Description
AddChall public)		
Parameters	Condition : List (String)	Initialises a new challeng condition from the para
Return values	n/a	Appends the new challen protected attribute.
CheckIfConditionMet (public)		
Parameters	Sequence : String	Returns True and sets th SetMet() if the Sequenc challenge, otherwise it re
Return values	Boolean	
ConvertConditionToString (private)		
Parameters	C : List (String)	Converts list of condition displaying on the screen parameter C, concatena ConditionAsString() us the delimiter.
Return values	ConditionAsString : String	
GetChall et (public)		
Parameters	Pos : Integer	Returns the Met status o Pos in the Challenges li
Return values	Boolean	

Identifier / Data		Description
GetLockDetails (public)		
Parameters	n/a	Used for displaying a challenge's c through the Challenges protected together the output string LockDe version of all the challenges for the been set to no
Return values	LockDetails: String	
GetLockSolved (public)		
Parameters	n/a	Returns the status showing if a loc through the Challenges protected there are any unmet ones, otherw
Return va	Boolean	
GetNumberOfChallenges (public)		
Parameters	n/a	Returns the number of Challenge number of challenges in this lock)
Return values	Integer	
SetChallengeMet (public)		
Parameters	Pos : Integer Value : Boolean	Uses the SetMet() method in the C attribute of a challenge at the posi list to Met or not Met using the Va
Return values	n/a	

Class: Card

Identifier / Data		Description
<<constructor>>		
Parameters	n/a	Initialises the CardNumber static attribute (class variab increments the static attribu NextCardNumber which m and updated for all objects o Initialises the Score protect
Return values	n/a	
GetCardNumber (public)		
Parameters	n/a	Returns the value of the pro
Return values	CardNumber : Integer	
GetDescription (public)		
Parameters	n/a	Returns the protected attrib string.
Return values	CardNumber: String	
GetScore (public)		
Parameters	n/a	Retu e protected attrib
Return values	Score : Integer	
Process (public)		
Parameters	Deck : CardCollection Board : CardCollection Hand : CardCollection Sequence : CardCollection CurrentLock : Lock Choice : String CardChoice : Integer	Base class method for the P classes to override.
Return values	n/a	

Class: ToolCard (inherits from Card)

Identifier / Data		Description
<<constructor>>		
Parameters	T : String K : String CardNo : Integer	Initialises the following protected attri • ToolT[illegible] from parameter T • [illegible] fr[illegible] parameter K [illegible]ardNumber from paramete
Return values	n/a	
		Invokes the SetScore() method to as base class for the ToolType.
<<constru[illegible]		
Parameters	T : String K : String	Initialises the following protected attri • ToolType from parameter T • Kit from parameter K
Return values	n/a	
		Invokes the SetScore() method to as base class for the ToolType.
GetDescription (public)		
Parameters	n/a	Overrides the GetDescription() meth return a concatenated string of the T for this ToolCard
Return values	String	
SetScore (public)		
Parameters	n/a	Assigns the c[illegible] Score from the p
Return values	n/a	

Class: [illegible]ultyCard (inherits from Card)

Identifier / Data		Description
<<constructor>>		
Parameters	n/a	Initialises the protected attri
Return values	n/a	Initialises CardNumber by c
<<constructor>>		
Parameters	CardNo : Integer	Initialises the protected attri
Return values	n/a	Initialises CardNumber from
GetDescription (public) <<override>>		
Parameters	n/a	Overrides the GetDescripti class to return the protected
Return valu[illegible]	[illegible]	

Process (public) <<override>>		
Parameters	Deck : CardCollection Discard : CardCollection Hand : CardCollection Sequence : CardCollection CurrentLock : Lock Choice : String CardChoice : Integer	Overrides the Process() me process the user choices fro user receives a difficulty car like to discard a key or 5 ca On ch ng the option to d le a key. This method th arameter is valid. ***Althoug*** ***errors in this check, AQA*** ***code is written as it was i*** If the Choice parameter co converted to an index by su a 'key' ToolCard in the play from the Hand and placed i If the Choice parameter do through deliberate user cho removed from the Deck and CardCollection.
Return values	n/a	

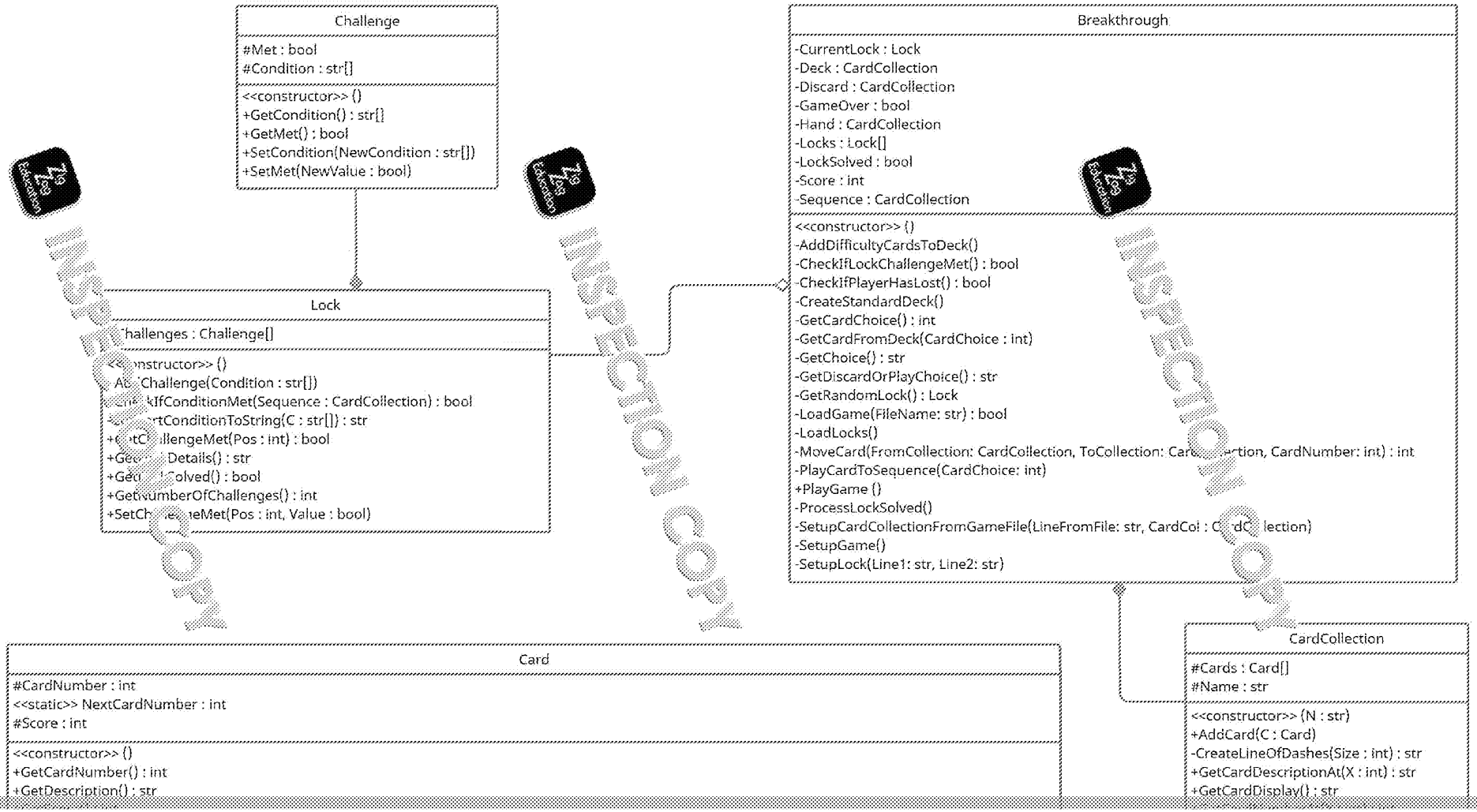
Class: CardCollection

Identifier / Data		Description
<<constructor>>		
Parameters	N : String	Initialises the following protected • Name from parameter N • Cards to an empty list
Return values	n/a	
GetCardDescriptio (public)		
Parameters	X : Integer	Returns a string containing the d X in the Cards list by invoking th method in Card.
Return values	String	
GetCardNumberAt (public)		
Parameters	X : Integer	Returns the CardNumber attribu the Cards list.
Return values	Integer	
GetName (public)		
Parameters	n/a	Returns the value of the protecte
Return values	Name : String	
AddCard (public)		
Parameters	C (Card)	Appends the value of parameter Cards.
Return values	n/a	
CreateLineOfDashes (public)		
Parameters	Size : Integer	Used in formatting a CardCollec
Return values	LineOfDashes : String	Returns an appropriately sized L of elements in a CardCollection CardCollection is greater than t Size).

Identifier / Data		Description
GetCardDisplay (public)		
Parameters	n/a	Used in formatting a CardCollec display output of a CardCollecti the collection Name and card de list attribu rds. If there are n co ame and 'empty' is re
Return values	CardDisplay : String	
		there are cards in the collectio which is either appropriately size the collection or is fixed at 10 if t collection is greater than 10. This fits correctly in the terminal wind It then uses indefinite iteration to using the GetDescription() met of the card at each element and and the | (pipe) symbol to create It then creates a second line of c underneath the 'line of cards' an
GetNumberOfCards (public)		
Parameters	n/a	Returns the number of cards in t Cards.
Return values	Integer	
RemoveCard (public)		
Parameters	CardNumber : Integer	Retur rd from Cards list re o t from Cards. If CardNumber is not a valid ind uninitialised variable CardToGe
Return values	CardtoGet : Card	
Shuffle (r		
Parameter	n/a	Uses definite iteration to perform from one random position to and attribute Cards in order to gener
Return values	n/a	

BREAKTHROUGH!

UML Class Diagram



BREAKTHROUGH

Theory Questions

These questions refer to the **Preliminary Material** and the **S**
but **do not** requi ny additional programmin

TOTAL MARKS: 80

1 Exami private method MoveCard. Currently this method return

(a) State a more appropriate name for this local variable.

.....

(b) Currently the MoveCard method returns an integer which represe that was moved. Sometimes this return value is ignored.

Evaluate the choice (of the programmer) to ignore the return valu return 0 in some cases, and suggest an alternative implementatio

.....

.....

.....

.....

.....

.....

.....

2 The class CardCollection currently contains an interface that expose structure of a list. For the sequence and the discard pile, a more appr would be either a queue or a stack.

(a) Justify whether you would use a queue or a stack. When giving y functionality of the data structure to the behaviour of the game.

.....

.....

.....

.....

.....

.....

.....

(b) In order to implement a stack or a queue for the sequence, justify (
thereof) that you would make to the inheritance structure.

.....

.....

.....

.....

(c) How would you design a new class to handle a CardCollection that
in[illegible] encapsulation?

.....

.....

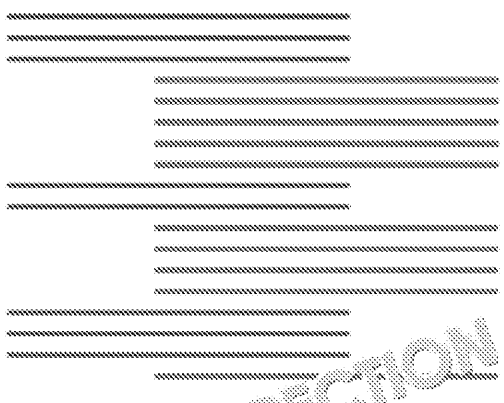
.....

.....

3 The Shuffle method of the CardCollection class currently swaps 10,0
cards in order to shuffle the deck.

Another way of shuffling the deck is to use a method that humans would normally use called a 'riffle shuffle'. This involves splitting the deck into two approximately equal piles and then flicking through each pile from the top down while combining the halves together into a single deck. Another way of thinking of this would be to imagine pushing the two halves together and having random number of cards between each card from each half they recombine.

For example, a deck combined from a blue half and an orange half might look something like this:



Note that in the perfect case a riffle shuffle would use one card from e
is not desired, and in reality, between 0 and 5 cards will normally inter
other half at any time.

a) Write a detailed algorithm for riffle shuffle in any format you choos
pseudocode, flow chart).

b) Explain the space complexity of your algorithm.

.....

.....

4 Examine the **CheckIfLockChallengeMet** method of the **Breakthroug**
CheckIfConditionMet method of the Lock class.

```
Lock:
Challenge Met: P c, F c, K c
Not met: P a, F a, P a
Sequence: P c, F c, K c, P a, F a, P a
```

(a) For the above sequence and lock, complete the trace table below
CheckIfLockChallengeMet method of the **Breakthrough** class.

Count	SequenceAsString	Return valu
	""	
5		

(b) If the above lock had a third challenge as below, then how would (please fill it in below)?

```
Not met: F a, P a
```

Count	SequenceAsString	Return value
	""	
5		

5 Examine the **ProcessLockSolved** method in the **Breakthrough** class methods called by that method.

(a) When a new lock is set, if that lock has been solved before, it will automatically replaced with a new lock the following turn (and trea just solved the first new lock) but reward the player for solving the

.....

.....

.....

.....

(b) D e logical change you would make to the code (no nee al you can) to ensure that this no longer happens.

.....

.....

.....

.....

6 Examine the **Shuffle** method in **CardCollection**. This method will ma of cards in the deck.

(a) Explain how the effectiveness and eff of this algorithm dec cards in the deck reduces.

.....

.....

.....

.....

.....

(b) Other than introducing a riffle shuffle, justify how you could impro efficiency of the algorithm by describing any changes below.

.....

.....

.....

.....

7 ToolC ca be instantiated with either two or three arguments.

(a) Explain what happens in the case where a third argument is supp where only two arguments are supplied.

.....

.....

.....

.....

.....

(b) State the purpose of a constructor.

.....

.....

8 Exami e classes Card, ToolCard and DifficultyCard.

(a) Using evidence from these classes in the program, explain the dif abstract and a concrete class.

.....

.....

.....

.....

.....

.....

.....

(b) Using evidence from the Card method, explain the difference betw
(static) and an attribute.

.....

.....

.....

.....

9 Find an [illegible] code for each of the following. Only write out the
line/s [illegible].

(a) Inheritance

.....

.....

(b) Aggregation association

.....

.....

(c) A dynamic data structure

.....

.....

10 This q[illegible] refers to the concept of polymorphism and how it is used

(a) Choose and then write out one or more lines of the skeleton progr
polymorphism and justify why this is an example of polymorphism

.....

.....

.....

.....

.....

.....

(b) Define the term '[illegible]orphism'.

.....

.....

.....

.....

11 A suggestion has been made to introduce a new AdvancedLock that challenge which is only revealed once the basic challenges have been

Explain the steps that you would take in order to do this, i.e. the logical change/addition and the reason for each step.

You are not required to implement this or to write any actual code.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

12 Examine the Process method in the DifficultyCard class and the PlayCardToSequence and GetCardFromDeck methods in the Breakthrough class.

Using the sequence below:

```
Not matched: P a, F a, K a
Sequence: P a, F a
Hand: P b, K a, F b, K c, P a
```

The player plays the 'K a' card to the sequence and then draws a difficulty card which will require them to either discard a key or five cards from the deck. The player will discard the 'K c' from their hand, which is currently in position 4.

Explain what will happen when the Process method is called under these circumstances, including specific references to the lines of code executed and in which order, and the values of variables, especially ChoiceAsInteger.

You will need to ensure that you look at the PlayCardToSequence and GetCardFromDeck methods in Breakthrough to be certain of the state of the Hand and Sequence when the DifficultyCard is drawn.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

13 The terms 'HAND', 'SEQUENCE', 'DECK' and 'DISCARD' all appear a
and in some cases more than once. This is an example of hard-coded
difficult to maintain and understand and also make it more prone to er

(a) Describe one method of avoiding hard-coding values that makes

.....

.....

.....

.....

(b) Explain why using hard-coded values makes the code more pron
understand.

.....

.....

.....

.....

14 Exception handling is used in several places in the skeleton code; two
the use of file handling.

(a) Describe why it is important to always use exception handling wh

.....

.....

.....

.....

(b) G example of another situation (not file handling) where exc
(it doesn't have to be from the skeleton code) and explain why.

.....

.....

.....

.....

15 This question refers to the PlayGame method of the Breakthrough c

Explain the use of the private attribute GameOver in this method, spe
is set and why it is used as the condition for iterative statements.

.....

.....

.....

.....

END OF QUESTIONS

BREAKTHROUGH

Theory Questions

These questions refer to the **Preliminary Material** and the **S**
but **do not** requi[illegible] ny [illegible]dditional programmin

TOTAL MARKS: 80

1 Exami[illegible] private method MoveCard. Currently this method return

(a) State a more appropriate name for this local variable.

(b) Currently the MoveCard method returns an integer which represe that was moved. Sometimes this return value is ignored.

Evaluate the choice (of the programmer) to ignore the return valu return 0 in some cases, and suggest an alternative implementatio

2 The class CardCollection currently contains an interface that expose structure of a list. For the sequence and the discard pile, a more appr would be either a queue or a stack.

(a) Justify whether you would use a queue or a [illegible]k. When giving y functionality of the data structure to t[illegible] el[illegible]iour of the game.

(b) In order to implement a [illegible] queue for the sequence, justify thereof) that you [illegible]ke to the inheritance structure.

(c) H[illegible]ating a new class to handle a CardCollection that im[illegible]encapsulation?

3 The Shuffle method of the CardCollection class currently swaps 10,0 cards in order to shuffle the deck.

Another way of shuffling the deck is to use a method that humans would normally use called a 'riffle shuffle'. This involves splitting the deck into two approximately even piles and then flicking through each pile from the bottom while combining the halves together into a single deck. Another way of thinking of this would be to imagine pushing the two halves together and having a random number of cards between each card from each half as they r[illegible]mbine.

For example, a deck combined from a bl[illegible] and an orange half might look somethi[illegible] is:

Note that in the p[illegible] a riffle shuffle would use one card from e is not [illegible]d, [illegible] reality, between 0 and 5 cards will normally inter other [illegible] any time.

a) Write a detailed algorithm for riffle shuffle in any format you choos pseudocode, flow chart).

b) Explain the space complexity of your algorithm.

4 Examine the **CheckIfLockChallengeMet** method of the **Breakthrough** **CheckIfConditionMet** method of the Lock class.

```
Lock:
Challenge Met: P c, F c, K c
Not met: P a, F a, P a
Sequence: P c, F c, K c, P a, F a, P a
```

(a) For the above sequence and lock, complete a trace table like the **CheckIfLockChallengeMet** method of the **Breakthrough** class.

Count	SequenceAsString	Return value
	""	
5		

(b) If the above lock had a third challenge as below, then how would (Complete an updated trace table)

```
Not met: F a, P a
```

Count	SequenceAsString	Return value
	""	
5		

5 Examine the **ProcessLockSolved** method in the **Breakthrough** class methods called by that method.

(a) When a new lock is set, if that lock has been solved before, it will automatically replaced with a new lock the following turn (and trea just solved the first new lock) but reward the player for solving the

(b) Describe the logical change you would make to the code (no nee although you can) to ensure that this no longer happens.

6 Examine the **Shuffle** method in **CardCollection**. This method will mak of cards in the deck.

(a) Explain how the effectiveness and efficiency of this algorithm dec cards in the deck reduces.

(b) Other than introducing a riffle shuffle, justify how you could impro efficiency of the algorithm by describing any changes below.

7 ToolCards can be instantiated with either two or three arguments.

(a) Explain what happens in the case where a third argument is supp where only two arguments are supplied.

(b) State the purpose of a constructor.

8 Examine the classes Card, ToolCard and DifficultyCard.

(a) Using evidence from these classes in the program, explain the dif abstract and a concrete class.

(b) Using evidence from the Card method, explain the difference bet (s and an attribute.

9 Find an example in the code for each of the following. Only write out th line/s of code.

(a) Inheritance

(b) Aggregation association

(c) A dynamic data structure

10 This question refers to the concept of polymorphism and how it is use

(a) Choose and then write out one or more lines of the skeleton prog polymorphism and justify why this is an exam of polymorphism

(b) Define the term 'polymorphism'.

11 A suggestion has been made to introduce a new AdvancedLock that challe hic only revealed once the basic challenges have been

Explain steps that you would take in order to do this, i.e. the logica change/addition and the reason for each step.

You are not required to implement this or to write any actual code.

12 Examine the Process method in the DifficultyCard class and the Pla GetCardFromDeck methods of the Breakthrough class.

Using the scenario below:

```
Not met: P a, F a, K a
Sequence: P a, F a
Hand: P b, K a, F b, K c, P a
```

The player plays the 'K a' card to the sequence and then draws a diffi require them to either discard or five cards from the deck. The p discard the 'K c' from hand, which is currently in position 4.

Explain what will happen when the Process method is called under th includi specific references to the lines of code executed and in whic values of variables, especially ChoiceAsInteger.

You will need to ensure that you look at the PlayCardToSequence an methods in Breakthrough to be certain of the state of the Hand and S the DifficultyCard is drawn.

13 The terms 'HAND', 'SEQUENCE', 'DECK' and 'DISCARD' all appear a
and in some cases more than once. This is an example of hard-code
difficult to maintain and understand and also make it more prone to er

(a) Describe one method of avoiding hard-coding values that makes

(b) Explain why using hard-coded values makes the code more pron
understand.

14 Exception handling is u in several places in the skeleton code; two
the use of file ha

(a) D why it is important to always use exception handling wh

(b) Give an example of another situation (not file handling) where ex
(it doesn't have to be from the skeleton code) and explain why.

15 This question refers to the PlayGame method of the Breakthrough c

Explain the use of the private attribute GameOver in this method, spe
is set and why it is used as the condition for two iterative statements.

END OF QUESTIONS

BREAKTHROUGH

Programming Tasks

These questions require you to load the **Skeleton Program** and to make

Note that any alternative or additional code changes that you deemed appropriate – ensuring that it is clear where in the Skeleton Program those changes

Task 1

This question refers to the PlayGame method of the Breakthrough class.

The number of cards left in the deck should be printed out after the current cards in the player's hand each turn.

Test the changes you have made:

Run the game and play two turns, showing the number of cards in the deck

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the PlayGame
- SCREEN CAPTURE(S) showing the required test

Task 2

This question refers to the PlayGame and GetChoice methods of the Bre
creation of a new attribute (with accessor methods), PeekUsed in the Loc

Introduce a **(P)eek** option. This can be used once per lock, and allows a p
see the next three upcoming cards. There should be a new command in P
the 'deck peek' is still available.

Create a new attribute in the Deck class called PeekUsed. Create access
to update and read PeekUsed attribute (get/set).

Update the GetChoice() method in the Breakthrough class to give the us
menu option should only appear if the PeekUsed attribute is False.

Introduce an option to the menu in the PlayGame() method to accept 'P' a
This menu option should only appear if the PeekUsed attribute is False. D
the deck using the GetCardDescriptionAt() method. Set the PeekUsed a
peek option has been chosen by the user.

When the player is given a new lock, set the PeekUsed attribute appropria
the peek option again.

Test the changes you have made:

Run the game and peek (peek is an option, it works) and then it's no longe
make sure it doesn't work even though the option isn't displayed. Solve a l
now an option again.

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the PlayGa
- PROGRAM SOURCE CODE showing changes made to the GetCho
- PROGRAM SOURCE CODE for the new PeekUsed attribute
- SCREEN CAPTURE(S) showing the required test.

Task 3

This question refers to the PlayCardToSequence method of the Breakth

Under the rules of the game, a player cannot play two cards of the same t
there is no error message warning the player when they attempt to do this

Modify the PlayCardToSequence method in the Breakthrough class to i
which tells the user that they cannot play two cards of the same type sequ

Use the GetCardDescriptionAt method to highlight to the user which card
and explain why it is the same as the type just played.

Test the changes you have made:

Run the game and show at least one turn played where the error does not
shows the new error message under the correct conditions of playing a du
show that (1) the error message is displayed and (2) the card is not played

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the PlayCar
- SCREEN CAPTURE(S) showing the required t

Task 4

This question refers to the PlayGame and GetChoice methods and the cr
attribute, MulliganUsed of the Breakthrough class.

Each player gets 1 'mulligan' per game where they can take all the cards i
discard pile and the sequence, put them together a shuffle up and deal
drawn (when repopulating the player's hand) should be sent to the discard
the current lock including any solved challenges will remain unchanged.

Create a new attribute for the Breakthrough class called MulliganUsed w
the Mulligan Used is False then display an additional **(M)ulligan** option ea
once the mulligan has been used, set the MulliganUsed attribute to True,
(M)ulligan option is no longer displayed or usable.

Test the changes you have made:

Run the game, solve one challenge, use mulligan, play one card to the se
attempt to mulligan again despite no menu option).

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the PlayGa
- PROGRAM SOURCE CODE showing changes made to the Breakt
- PROGRAM SOURCE CODE showing changes made to the GetCho
- SCREENSHOT(S) showing the required test

Task 5 D

This question refers to the PlayGame and GetChoice methods of the Bre

The player will have a new option in PlayGame to **(Q)uit**, and for this they
score for each card remaining in the deck. Print out their final score as the

Note that the code should exit cleanly/nicely without using any Applicatio
type statements or GoTo state[illegible]ough break/continue are allowed

Test the c[illegible]s you have made:

Play one turn of a game, choose quit.

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the PlayGa
- PROGRAM SOURCE CODE showing changes made to the GetCho
- SCREEN CAPTURE(S) showing the required test

Task 6 Diffi

This question refers to the GetCardFromDeck method of the Breakthrou
new method, DisplayStats, modifying two existing methods, AddCard an
adding three new attributes, NumPicks, NumFiles and NumKeys, in the

Introduce a stats / card count to the CardCollection class which keeps tra
out of the deck and calculates the % chance that the next card tile in the d

Introduce three new attributes to the CardCollection class called NumPic
which will be updated every time a ToolCard is added to or removed from

Create a new method in the CardCollection class called DisplayStats. T
the percentage chance of the next card being a key, pick or file based on t
the number of cards left in the deck.

When the player receives a difficulty card, use the DisplayStats method to
GetNumberOfCards method in the CardCollection class to display the fo
they choose 'lose a key or discard 5 cards from the deck'.

*There is a X% chance that the next card will be a key, a Y% chance that it
that it will be a pick.*

The percentages should be displayed to two decimal places.

Replace X, Y and Z with the appropriate values. Note that they will not nor
because there are also difficulty cards in the deck.

Test the changes you have made:

Run the game until a difficulty card is drawn and show the printout of the s
(after the hand and before asking which card).

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the GetCar
- PROGRAM SOURCE CODE showing changes made to the CardCo
- SCREEN CAPTURE(S) showing the required test

Task 7 Diffi

This question involves the CreateStandardDeck, ProcessLockSolved a methods of the Breakthrough class, as well as the creation of a new SetC Card, ToolCard and CardCollection classes.

Introduce three new 'multi-tool' cards – a **multi-[illegible]**), a **multi-key (K**

At the start of a standard game (rather than loading a save game file), the de of these new types of card multi-tool cards can be dealt to the player's ha cards are.

On playing [illegible]ti-tool card, the player should be given the option to choos assign the card to before it is added to the sequence, therefore allowing a any lock challenge of that type.

When a lock has been solved, three new multi-tool cards (one of each type available for the next lock and the deck is reshuffled (as normal).

Test the changes you have made:

Play the game and show the use of at least one multi-tool card, the print s sequence both before and after the multi-tool is played.

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the CreateS
- PROGRAM SOURCE CODE showing changes made to the Process
- PROGRAM SOURCE CODE showing changes made to the PlayCar
- PROGRAM SOURCE CODE for the new SetCardToolkit method (in CardCollection classes)
- SCREEN CAPTURE(S) showing the required test.

Task 8 Diffi

This question refers to the GetLockDetails method of the Lock class and Breakthrough class.

Challenges are to be marked as 'partially met' (rather than just 'met' or 'no solved. A challenge is partially met if the end of the sequence (last one or an unsolved challenge.

Modify the call to GetLockDetails from PlayGame to pass in the sequenc

Modify GetLockDetails so that if the challenge is not met then it checks to For challenges of three cards, only check the last two cards and it become of the sequence matches the first card of the challenge or the second last the first card of the challenge and the last card of the sequence matches th challenge.

In general, check N-1 cards where N is the number of cards in the challen challenges of one card cannot be partially met. You only need to solve the three cards exactly.

Test the changes you have made:

Run the game and play one card to the sequence that doesn't match any one towards one of the three card challenges that matches the first card fo screen showing this entire turn.

Then play a second card to sequence that matches the second card o

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the PlayGa
- PROGRAM SOURCE CODE showing changes made to the GetLoc
- SCREEN CAPTURE(S) showing the required test

Task 9 Diffi

This question refers to the PlayGame method of the Breakthrough class.

Introduce a bonus for solving locks using fewer cards. Once the first card is sequence for a new lock, a counter starts and one is added every time a pl (discarding or playing to the sequence).

Once a lock is solved (all the chains are), a player receives an extra point f the counter, after which it is reset. The player simply receives 0 if the count message c[illegible] bonus points that were awarded (including 0 if that

Test the changes you have made:

Run the game and play two locks, one solved in under 20 cards to show a in over 20 cards to show a bonus score of 0.

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the PlayGa
- SCREEN CAPTURE(S) showing the required test

Task 10 Diffi

This question refers to the ProcessLockSolved, SetupGame and GetCa as the creation of a new method, AddGeniusCardToDeck of the Breakth of a new class called GeniusCard.

Introduce a new 'Genius Card' which is added to the deck at the start of a lock chance of having a 'Genius Card' in a deck.

A player can choose to use the 'Genius Card' when they draw it to solve a ask which card will be discarded and then reshuffled into the de from the discard pile.

Note that if a GeniusCard is drawn when filling up the hand it should be d message should be printed to this effect.

Create a method called AddGeniusCardToDeck which has a 25% chanc to the deck. This should be called from ProcessLockSolved and SetupG

Create a new class for the GeniusCard which inherits Card with CardTyp the GetCardFromDeck method of Breakthough to ensure that the card is drawn.

Test the changes you have made:

Run the game and play until a 'Genius Card' is drawn, then choose yes an challenge in the current lo

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the Proces
- PROGRAM SOURCE CODE showing changes made to the SetupG
- PROGRAM SOURCE CODE showing changes made to the GetCar
- PROGRAM SOURCE CODE for the new GeniusCard class
- PROGRAM SOURCE CODE for the new AddGeniusCard method
- SCREEN CAPTURE(S) showing the required tes

Task 11 Di

This question refers to the addition of a new attribute in the Breakthrough cl
GetCardFromDeck method of the Breakthrough class as well as the creat
PrintToolsAvailable, for the CardCollection class.

Introduce the concept of 'Buying a tool' from the dec

Add a new attribute, Credits, to the Br ough class which contains the
currently has. At the start of the gan s player has 10 credits. When a play
sequence or discarded c t they have at least 2 credits remaining, they s
like to buy (y/ ore their hand is refilled from the deck. If they choos
card as no t otherwise the new card will be the tool card that they purc

Players can 'buy' a 'Key' card at the cost of 3 credits, and 'file' or 'pick' cards a
When the player chooses 'y' to buy a tool, they should be prompted with the f
have 0 availability should not be listed).

```
1.    F a (1 available)
2.    F b (1 available)
3.    F c (1 available)
4.    P a (1 available)
5.    P b (1 available)
6.    P c (1 available)
7.    K a (1 available)
8.    K b (1 available)
9.    K c (1 available)
10.   No Tool (buy nothing)
```

***Note**: the actual number a*

***Note**: keys (items 7–9) sho*
at least 3 credits left. All m
numbers given above even
e.g. item 10 should always
the player changed their m

The new PrintToolsAvailable meth s d take one parameter, KeysAv
player has at least 3 credits e otherwise is False. It should return an array
available tool card a ype, for example, if the deck contains three files fr
at index 3 deck, no files from toolkit b and one file from toolkit c which is
the first par e array returned would look like this:

```
[ 3, -1, 12, ...
```

***Note:** -1 is used to indicate*

Test the changes you have made:

1. Run the game and play any card to the sequence, then choose 'y' whe
a tool. Select any tool listed as available, play it to the sequence and t
asked if you would like to buy a tool; show all the output produced incl
and the tool card being added to the player's hand each time.
2. Continue playing the game and buying tools until you have spent a tota
pick/file and 2 keys) and then show the printed li tools available wh

Evidence that you need to p

- PROGRAM SOURCE CODE showing changes made to the GetCardF
- PRO SOURCE CODE for the new Credits attribute
- PROGRAM SOURCE CODE for the new PrintToolsAvailable method
- SCREEN CAPTURE(S) showing the required test

Task 12

This question refers to CheckIfLockChallengeMet method of the Breakth

Create an 'Advanced' mode where, for any challenge that requires three c once the challenge is solved move the cards used to solve it from the seq exposing the previous card on the sequence, which could then possibly be challenge.

For example, if the sequence contains:

```
Fa, Kc,
```

and the current challenge is Pb, Kb, Fb. Suppose you play Kb and Fb to th current challenge but instead of the sequence extending to:

```
Fa, Kc, Pb, Kb, Fb
```

it will be contracted to:

```
Fa, Kc
```

and the Pb, Kb and Fb cards from the challenge that was just solved will b

Test the changes you have made:

Run the game and restart until you get a Lock and at least one challenge o of three cards. Play until you solve the three card challenge and then play challenge. The screen capture(s) should show the Lock, Sequence and Ha card to solve the three card challenge and the Lock and Sequence after yo

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the CheckIfLo
- SCREEN CAPTURE(S) showing the required test

Task 13

This question refers to PlayGame and GetChoice methods and to the cre
method in the Breakthrough class. It also requires the creation of new Ge
GetChallengesMetAsString methods in the Lock class.

The PlayGame menu should have a **(S)** ve option which will save the gam
allow it to be reloaded (from the main menu when you first start the game).

In order to understand the format of the save game file, you will have to in
LoadGame method of the Breakthrough class.

Print out a suitable message stating whether the game was saved succes

Test the changes you have made:

1. Take a copy of the **game1.txt** file and rename it **backup.txt**.
2. Run the game until you get a lock with at least two challenges. Sol
save the game as '**game1.txt**' (it shouldn't prompt you). Load the g
of it has been correctly restored.
3. Restore the original **game1.txt** from **backup.txt**.

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the PlayGa
- PROGRAM SOURCE CODE showing changes made to the GetCho
- PROGRAM SOURCE CODE for the new SaveGame method
- PROGRAM SOURCE CODE for the new GetChallengesAsString m
- PROGRAM SOURCE CODE for the new GetChallengesMetAsStri
- SCREEN CAPTURE(S) showing the required test

Task 14

This question refers to PlayGame and PlayCardToSequence methods an attribute, BonusPool, in the Breakthrough class. It also requires the creat IsPartial, in the Lock class, which takes Sequence as a parameter as wel GetCardDescriptionAt method.

Introduce a bonus for playing consecutiv ar to a lock that solves a cha card played in a row that goes ow solving a challenge will add 5 to th pool will be added to for each card. The GetCardDescriptionAt empty stri e index doesn't exist.

For example, the bonus pool is 0 and a player plays a card towards challen added to their score along with their normal score and the bonus pool is in does anything except play another correct card towards challenge 1, then 0; otherwise they will get the score for the card played as normal, plus the bonus pool will be increased to 10 and so on.

Test the changes you have made:

Run the game and keep discarding until you have all three cards required solve it one card after another; continue playing and play a card to a challe sequence that is not part of the challenge.

Evidence that you ne ide:

- PRO M CE CODE showing changes made to the PlayGa
- PRO M SOURCE CODE showing changes made to the PlayCa
- PROGRAM SOURCE CODE showing changes made to the GetCar
- PROGRAM SOURCE CODE for the new BonusPool attribute
- PROGRAM SOURCE CODE for the new IsPartial method
- SCREEN CAPTURE(S) showing the required test

Task 15 Diffic

This question refers to the ProcessLockSolved, GetCardFromDeck and methods, and to the creation of new private GenerateSolubleLock and G and a new private attribute FinalLock in the Breakthrough class. It also r public IsSoluble method in the Lock class that takes in Deck and Hand

EXTRA FILE NEEDED: **game2.txt**

Every lock generated must be solvable based on the cards left in the deck exhaust the deck. If the lock cannot be solved, then choose a new random in a row (without a suitable lock being found) then display a message 'Fina generate a lock with two challenges that can be solved.

Once those challenges are solved, there should be a message from Chec instead of saying the player lost, prints out 'You have solved the final lock.

When approaching this task you should ignore the effect of Difficulty card check that the Deck and Hand combined contain the requisite number of lock.

The attribute FinalLock should be set to 0 at the start and then set to 1 in the final lock is set. When CheckIfPlayerHasLost runs it should set Fina that the final turn is played). If FinalLock is 1 and there are no cards left in doesn't lose until all the cards from the hand are gone.

Test the changes you have made:

1. Change the game to load the file **game2.txt** instead of **game1.txt** load game.
2. Play the game until the message 'Final Lock' is displayed, then sol final turn.

Evidence that you need to provide:

- PROGRAM SOURCE CODE showing changes made to the ProcessL
- PROGRAM SOURCE CODE showing changes made to the CheckIfP
- PROGRAM SOURCE CODE showing changes made to the GetCardF
- PROGRAM SOURCE CODE for the new GenerateSolubleLock meth
- PROGRAM SOURCE CODE for the new GenerateChallenge method
- PROGRAM SOURCE CODE for the new IsSoluable method
- PROGRAM SOURCE CODE for the new FinalLock attribute
- SCREEN CAPTURE(S) showing the required test

BREAKTHROUGH

Possible Additional Programmin

1. Create an extra toolkit (e.g. 'd') and ad[illegible] involving this to the l
2. Introduce a *Swiss Army K*[illegible] which can be used as any singl toolkit.
3. Ad[illegible] ful[illegible]ity to allow a player, at the start of a turn, to (U)n rev[illegible]rything – including the score, sequence, hand and discar the start of the previous turn. There should be one undo available p possible to use it on the first turn of a new lock.
4. Add a *High Scores* file and ability to view this from a main menu.
5. Add levels so that different locks have different challenges which w depending on the current level. This could be linked to (11 – comp number of toolkits used, e.g. 2, 3 or 4.
6. Add a *Mighty Hammer* card that can smash (solve) the current lock your hand and play it later.
7. Introduce a user-defined locks option. This generates a rough pseu one player can choose a lock sequence and another has to try to u original game of *Mastermind*). A user-defined lock must follow the at least two must be files, and at least one must be a pick.
8. Introduce a second type of lock [illegible] Maths Lock', whereby the play the way they are now and [illegible] new maths locks. This will involve each card call[illegible] 'number'. The value of 'number' is displayed in a (pi[illegible] currently displayed. Cards can be used for their ma the[illegible] attribute. For example, if a lock contains four files – each that gives a total lock value of 20. The player needs to play a sequ a total value of 20. For example, if the player plays two picks, each then the lock will open. These new 'Maths Locks' are solved only u are independent of the tool type and tool kit.
9. Receive a bonus of 50 if you quit and the current challenge could n and deck as they are currently).
10. Add an *Autoplay* mode which shows a computer simulation of the g
11. Design a formula to compute a complexity value for a lock.
12. Validation of card to play (with exception handling) for choosing wh response to a difficulty card.
13. Validation on entry of choice (or [illegible] so that the player can o
14. Be able to sacrifice a card (removed from the game) in order to cha
15. Exam[illegible] .txt file (or game) closely and draw a flow diagr pla[illegible] make to move from the 'saved state' to 'end of the gam look[illegible] the data in the game1.txt file rather than playing the gam

BREAKTHROUGH!

ory Questions (Mark Scheme)

Question		Sug		Total Marks	Marking Guidance
1	(a)	e.g.		1 mark	**A:** Similar names with meaning to explain the score. **R:** Spaces in names. **I:** case.
	(b)	**4 ma** creat Scor from **Exar** **3 ma** playe **2 ma** **1 ma**	ctice **[1]**… One alternative would be to re **[1]**… which takes the current player core **[1]**… and to remove the return value rks: ove the logic to the place where t ne the score is needed **[1]** separate getScor method **[1]**. g it is bad practice **[1]**.	4 marks	**A:** any reasonable suggestion. **A:** answers without passing Score as a eter and dealing with the extra score now. **A:** answers where there is a scoring method in CardCollection which 'knows' whether to score or not. **A:** passing score in by reference and having a new attribute on CardCollection to indicate if a card added/played should affect the score.
2	(a)	The is a L the d to th or th wher disca	he end and taken from the same end which ructure that would be appropriate **[1]**… For the whole stack but generally just play cards k in **[1]**… a stack could be suitable for this t is easy to iterate through and print out k for the sequence and no change for the	4 marks	1 mark for each point (MAX. 4) **A:** stack for discard pile. **R:** queue for either.
	(b)	Crea from stac end	or SequenceCardCollection that inherits des methods to ensure that it behaves as a hat you can only remove the card at the	2 marks	1 mark for each point (MAX. 2) **A:** any suitable method for a stack, not just isEmpty()
	(c)	It wo empl know	uence you could just remove a card until it's would be hidden **[1]**… avoiding the need to what constitutes the start or end **[1]**.	2 marks	for each point (MAX. 2)

Question		Sugg		Total Marks	Marking Guidance
3	(a)	Splitt numi coun the c cards fully	rs to do the same thing: choosing a he combined deck. A circular deck with cards from the other half and add them to r of cards (0 to 5, A. 1 to 5) **[1]**… Taking an the top **[1]**… Repeating until the deck is	6 marks	1 mark for each point
	(b)	Most befor	ce that of the random shuffle that existed ed/combined deck.	1 mark	**A:** any version of the idea for 1 mark. **A**: circular deck solutions with space complexity of the same as the storage for the deck as long as they are explained properly.
4	(a)	Cou 5 4 3		5 marks	1 mark for the Count column (**I:** spaces) 1 mark for a final return value of True 1 mark for the first value in the SequenceAsString column 1 mark for the last value in the SequenceAsString column 1 mark for the correct middle values in the SequenceAsString column (between the first and last value) **DPT:** -1 only for a missing space
	(b)	Cou 5 4		3 marks	1 mark for each column **DPT:** -1 only for a missing space (note that this is across parts (a) and (b) combined, total of -1 for a missing space across the two parts).
5	(a)	Whe attrib **[1]**… comr settir remo	by changing the value of the protected , these challenges are not reset to False sen it will appear but the moment that a alled which will mark the lock as solved by ough class to True **[1]**… The lock is not om lock is chosen when it is solved **[1]**.	2 marks	1 mark for each point (MAX. 2)

Question		Sugg		Total Marks	Marking Guidance
5	(b)	**Eithe** Once wher **Or:** Once *Fals*	ocks so that it cann t collected again llenges and set the attribute *Met* for each to	2 marks	2 marks available for either solution as long as the details are clearly explained, otherwise award 1 mark **A:** any other reasonable solution including the idea of moving *LockSolved* into the *Lock* class and checking it when choosing a new lock.
6	(a)	As th arrar **[1]**… there exce	hen there will be more swaps than possible itional swaps redundant and inefficient o combinations and with 6 only 720 but e chance of the algorithm causing an for this **[1]**.	3 marks	Award 1 mark for each point **A:** any expression of the idea that 10000 is more swaps than you need, which makes the extra ones unnecessary for the first mark. **A:** any reference or example to decreasing combinations for the second mark.
	(b)	The wher This value	k size or could be set to a lower threshold nt and setting a swaps variable to a lo e deck is large **[1]**.	2 marks	Award 1 mark for each point **A:** any expression of each concept for each mark. **A:** any other reasonable suggestions that would give 10000 for large decks and a much lower number for small decks.
7	(a)	Whe **[1]**… varia third set b	to set the *ToolType* and *Kit* respectively ilable *CardNumber* from the class/static ctor when it is called **[1]**… In the case of a ctor is not called and the *CardNumber* is	3 marks	1 mark for each point (MAX. 3)
	(b)	A co whic	utes appropriately and return an object	1 mark	**A:** to instantiate the class with the correct values.
8	(a)	An a An e A co the T	ld never create an instance of **[1]**… ty *Process* method **[1]**. an instance of **[1]**… An example of this is	4 marks	For each type of class, there needs to be a description for the first mark with an example for the second mark. For the concrete class, pretty much any class from *Breakthrough* onwards in the program (*Card*) can be given as an example.

Question		Sugg		Total Marks	Marking Guidance
8	(b)	A cla wher each	every object and is cha[illegible] in them all art off the same bu[illegible] different value in he others **[1]**.	2 marks	1 mark for each point
9	(a)	Inher	nd behaviours/methods of its parent.	1 mark	**A:** other words with similar meanings.
	(b)	Aggr linke	ns another class but their lifespans are not	1 mark	**R:** answers that do not refer to lifespan in some way.
	(c)	A da stora	shrink and grow over time according to the	1 mark	

10	(a)	***Solu***
		Code
		`Move`
		… be
		whicl
		actua
		Solu
		Code
		`Publ`
		`Card`
		`Card`
		`Card`
		… be
		will n
		Solu
		Code
		`If D`
		… be
		attrib
		that r
			`rAt(CardChoice - 1))`
			ds and ToolCards **[1]**… as Cards **[1]**…
			he statement resolves any methods will
			`s CardCollection, ByVal Discard As`
			`ion, ByVal Sequence As`
			`, ByVal Choice As String, ByVal`
			1]… in the parent Card class **[1]**… which
			cards but behave as themselves **[1]**.
			`hen`
			parent) **[1]**… but when the call resolves the
			tor for DifficultyCard **[1]**… which means
			ltyCard (morph to child) **[1]**.
				4 marks	For any one possible answer: 1 mark for each point and 1 mark for the code
					A: any example of code related to inheritance for 1 mark provided the explanation gains at [illegible] mark.
					R: [illegible]de only with no explanation.

	(b)	Inter the n	parent **[1]**… but having the object resolve nheritance structure from the child **[1]**.	2 mar[illegible]	1 [illegible]rk for each point **A:** execute the method in the child.

Question		Sugg		Total Marks	Marking Guidance
11		Crea GetL final a priv time	nerits from Lock **[1]**… and override the basic challenges a[illegible] it unlocks the **[1]**… this will then mean that it can refer to when checking GetLockSolved the next	6 marks	1 mark for each point
12		The ' will b as M hand now Choi in Ch Choi state that to th	GetCardFromDeck is called, CardChoice eady been moved from the hand to the deck e before GetCardFromDeck **[1]**… The 'K c' as the key to discard because it is CardChoice is 2 and Choice is 3 **[1]**… ange checked (1–5) successfully and stored nteger is decremented by 1 **[1]**… and as 1 again to give 1 **[1]**… The selection index ChoiceAsInteger (which is 1) and]… and 5 cards are discarded from the deck	8 marks	1 mark for each point (MAX. 8)
13	(a)	Usin chan	once at the top of program and could be	[illegible]	1 mark for each point
	(b)	It is p **[1]**…	e wrong values **[1]**,… or mis[illegible] e easier to understand **[1]**	2 marks	1 mark for each point (MAX. 2) **A:** opposite points.
14	(a)	File f remo	]… because files could be locked **[1]**…	2 marks	1 mark for each point (MAX. 2)
	(b)	Conv and a	because if it fails you want to catch the error	2 marks	1 mark for each point **A:** any example of validation. **A:** any reasonable example including custom exceptions.
15		It is u cards value exit t	se it needs to exit if the player runs out of the outside loop it is based on the return ks if the deck has 0 cards so this needs to	2 marks	1 mark for each point **A:** answers that don't refer to CheckIfPlayerHasLost as long as they refer to where GameOver is set.

BREAKTHROUGH

Programming Tasks (Mark Sch

Task 1

Coding

- Printing out the number of cards left in the deck correctly each turn [1 mark]

Example Solution

```
Console.WriteLine(Sequence.GetCardDisplay())
' Code added
Console.WriteLine("There are " & Deck.GetNumberOfCards() & "
' End of addition
Console.WriteLine(Hand.GetCardDisplay())
```

Testing:

- Printing out the number of cards left correctly between SEQUENCE and HAND [1

```
SEQUENCE: empty


There are 33 cards left in the deck.

HAND:
```

Task 2

Coding:

- Changing GetChoice to show Peek (even if it doesn't check GetPeekUsed) [1 m
- Changing PlayGame to accept 'P' and printing out the three cards in the deck (reg [1 mark]
- Adding the PeekUsed attribute with get's and sets to Lock [1 mark]

Example Solution

Changes to GetChoice

```
            Console.WriteLine()
            ' code change
            If CurrentLock.GetPeekUsed() Then
                Console.Write("(D)iscard inspect, (U)se card:> ")
            Else
                Console.Write("(D)iscard inspect, (U)se card, (P)eek:> "
            End If
            ' end change
            Dim Choice As String = Console.ReadLine().ToUpper()
```

Changes to PlayGame

```
            Console.WriteLine(Discard.GetCardDisplay())
        ' code added
        Case "P"
            If Not CurrentLock.GetPeekUsed() Then
                Console.WriteLine("The next three cards are: " & Deck.Get
                Deck.GetCardDescription(1) & " " & Deck.GetCardDescrip
                CurrentLock.SetPeekUsed()
            End If
        ' end addition
        Case "
```

Changes to Lock

```
        ' code added
        Private PeekUsed As Boolean = False
        ' end addition...

        'code added
        Public Overridable Function GetPeekUsed() As Boolean
            Return PeekUsed
        End Function

        Public Overridable Sub SetPeekUsed()
            PeekUsed = True
        End Sub
        'end addition
```

Testing:

- Peek is an option, works correctly and then disappears [1 mark] ↓

```
(D)iscard inspect, (U)se card, (P)eek:> p
The next three cards are: F a, F c, K a

Current score: 1

CURRENT LOCK
------------
Not met:        P c, F c, K c


SEQUENCE:

------
| P c |
------


HAND:

----------------------------
| P a | K b | P b | P a | K b |
----------------------------


(D)iscard inspect, (U)se card:> u
Enter a number between 1 and 5 to specify card to use:> 1
(D)iscard or (P)lay?:> d

Current score: 1
```

- Peek reappears for the next lock and works until the lock disappears and doesn't work

```
(D)iscard inspect, (U)se card:> u
Enter a number between 1 and 5 to specify card to use:> 5
(D)iscard or (P)lay?:

A challenge lock has been met.

Lock has been solved.  Your score is now: 21

Current score: 21

CURRENT LOCK
------------
Not met:        P a, F a, P a
Not met:        P c, F c, P c


SEQUENCE:

-----------------
| P c | F c | K c |
-----------------


HAND:

----------------------------
| K b | K b | K a | F a | F
----------------------------


(D)iscard inspect, (U)se card, (P)eek:> p
The next three cards are: Dif, P b, F b

Current score: 21
```

```
(D)iscard inspect, (U)se card

Current score: 21

CURRENT LOCK
------------
Not met:        P a, F a, P a
Not met:        P c, F c, P c
```

Task 3

Coding:

- Checking for correct condition to print out the error [1 mark]
- Printing out a sensible error message with the card or type that is in error [1 m

Example Solution

Changes to `PlayCardToSequence`

```
            dFromDeck(CardChoice)
            ' code added
        Else
            Console.WriteLine("Error: The card you are trying to
            Hand.GetCardDescriptionAt(CardChoice - 1) & ") is th
            the sequence.")
            ' end addition
        End If
```

Testing:

- Showing the error message and the hand and sequence afterwards confirming tha discarded [1 mark] ↓

```
SEQUENCE:

------
| P a |
------

HAND:

---------------------------
| P b | F c | F a | F b | F
---------------------------

(D)iscard in (U)se card:> u
Enter a number between 1 and 5 to specify card to use:> 1
(D)iscard or (P)lay?:> p
ERROR: The card you are trying to play (P b) is the same type as the last card in the sequ

Current score: 1

CURRENT LOCK
------------
Not met:      P b, K b, F b


SEQUENCE:

------
| P a |
------

HAND:

---------------------------
| P b | F c |
---------------------------

(D)iscard inspect, (U)se card:>
```

Task 4

Coding:

- Printing out the correct message only when a mulligan is available [1 mark]
- Adding the MulliganUsed attribute to Breakthrough and initialising it to False [1
- Implementing the mulligan to add all the cards from the player's hand, the discard [1 mark]
- Shuffling up and dealing again (and discarding any difficulty cards drawn) [1 mark]

Example Solution

Changes to GetChoice

```
            Console.WriteLine()
            ' code changed
            If MulliganUsed Then
                Console.Write("(D)iscard inspect, (U)se card:> ")
            Else
                Console.Write("(D)iscard inspect, (U)se card, (M)ulligan
            End If
            ' end change
            Dim Choice As String = Console.ReadLine().ToUpper()
```

Changes to Breakthrough

```
        Private LockSolved As Boolean
        ' code added
        Private MulliganUsed As Boolean = False
        ' end addition
```

Changes to PlayGame

```
                    "D"
                        Console.WriteLine(Discard.GetCardDisplay
                        'code added
                    Case "M"
                        If Not MulliganUsed Then
                            Dim Count As Integer
                            ' move cards from sequence to deck
                            For Count = 1 To Sequence.GetNumberO
                                MoveCard(Sequence, Deck, Sequenc
                            Next
                            ' move cards from discard pile to de
                            For Count = 1 To Discard.GetNumberOf
                                MoveCard(Discard, Deck, Discard.
                            Next
                            ' move cards from hand to deck
                            For Count = 1 To Hand.GetNumberOfCar
                                MoveCard(Hand, Deck, Hand.GetCar
                            Next
                            ' shuffle up and deal
                            Deck.Shuffle()
                            For Count = 1 To 5
                                While Deck.GetCardDescriptionAt(
                                    MoveCard(Deck, Discard, Deck
                                End While
                                MoveCard(Deck, Hand, Deck.GetCar
                            Next
                            MulliganUsed = True
                        End If
                        'end addition
                    Case "U"
```

Testing:

- Showing a mulligan being used after solving a challenge [1 mark] ↓

```
CURRENT LOCK
------------
Not met:        P a, F a, P a
Not met:        P b, F b, P b
Challenge met: K c


SEQUENCE:

------
| K c |
------


HAND:

-----------------------------
| K a | F b | P b | K a | P b |
-----------------------------


(D)iscard inspect, (U)se card, (M)ulligan:> m

Current score: 8

CURRENT LOCK
------------
Not met:        P a, F a, P a
Not met:        P b, F b, P b
Challenge met: K c


SEQUENCE:


HAND:

-----------------------------
| K b | F a | K a | P b | F c |
-----------------------------


(D)iscard inspect, (U)se card:> ▯
```

- Showing an attempt to use the mulligan again failing [1 mark] →

Task 5

Coding:

- Printing out quit as a menu option and including it in the selection statement in Pl
- Cleanly exiting the main game loop in PlayGame without u[illegible] Application.ex[illegible] mechanism and successfully ending the program [illegible]

Example Solution

Changes to GetChoice

```
        Co[illegible]e.[illegible]eline()
        [illegible]change
        Co[illegible]le.Write("(D)iscard inspect, (U)se card, (Q)uit:> ")
        ' end change
        Dim Choice As String = Console.ReadLine().ToUpper()
```

Changes to PlayGame

```
    Public Sub PlayGame()
        Dim MenuChoice As String
        'code added
        Dim HasQuit As Boolean = False
        'end addition
        If Locks.Count > 0 Then
            GameOver = False
            CurrentLock = New Lock()
            SetupGame()
            While Not GameOver
                LockSolved = False
                'code change
                While Not LockSolved [illegible] GameOver And Not HasQui
                    'end chang[illegible]
                    Con[illegible] Wri[illegible]eline()
                    [illegible].WriteLine("Current score: " & Score)
                    [illegible]nsole.WriteLine(CurrentLock.GetLockDetails())
                    Console.WriteLine(Sequence.GetCardDisplay())
                    Console.WriteLine(Hand.GetCardDisplay())
                    MenuChoice = GetChoice()
                    Select Case MenuChoice
                        Case "D"
                            Console.WriteLine(Discard.GetCardDisplay(
                            'code added
                        Case "Q"
                            HasQuit = True
                            'end addition
                        Case "U"
                            Dim CardChoice As Integer = GetCardChoic
                            Dim DiscardOrPlay As String = GetDiscard
                            If DiscardOrPlay = "D" Then
                                MoveCard(Hand, Discard, Hand.GetCard
                                GetCardFromDeck([illegible] Choice)
                            ElseIf Discar[illegible]la[illegible] = "P" Then
                                Pl[illegible]TC[illegible]quence(CardChoice)
                            [illegible]
                    End [illegible]ct[illegible]
                    [illegible]urrentLock.GetLockSolved() Then
                        LockSolved = True
                        ProcessLockSolved()
                    End If
                End While
                'added the If and moved the existing code to the els
                If HasQuit Then
```

```
                GameOver = True
                Score += Deck.GetNumberOfCards()
                Console.WriteLine("Final score: " & Score)
            Else
                GameOver = CheckIfPlayerHasLost()
            End If
            'end change
        End While
```

Testing:

- Printing out a final score of 1 (if a Pick was played), 34 (if a File was played) or 33

```
HAND:

-----------------------------
| P b | P a | F c | K b | K c |
-----------------------------


(D)iscard inspect, (U)se card, (Q)uit:> u
Enter a number between 1 and 5 to specify card to use:> 2
(D)iscard or (P)lay?:> p

Current score: 1

CURRENT LOCK
------------
Not met:       P a, F a, P a
Not met:       K b


SEQUENCE:

------
| P a |
------


HAND:

-----------------------------
| P b | F c | K b | K c | F a |
-----------------------------


(D)iscard inspect, (U)se card, (Q)uit:> q
Final score: 33
```

Task 6

Coding:

- Adding the three attributes numPicks, numKeys and numFiles to the CardColle
  0 [1 mark]
- Ensuring that at least one attribute is updated corr w n a card is added [1 m
- Ensuring that all three attributes are updated re th when a card is removed [1
- Creating a DisplayStats method ll out the percentage of each type
  if not to two decimal places) th 'correctly' means dividing the number of th
  in the deck. [1 mark]

Example Solu

Changes to the CardCollection class

```
' code added
Private NumPicks As Integer = 0
Private NumFiles As Integer = 0
Private NumKeys As Integer = 0
'end addition

 Public Sub AddCard(ByVal C As Card)
     'code added
     If C.GetDescription()(0) = "F" Then
         NumFiles += 1
     ElseIf C.GetDescription()(0) = "P" Then
         NumPicks += 1
     ElseIf C.GetDescription()(0) = "K" Then
         NumKeys += 1
     End If
     'end addition
     Cards.Add(C)

Public Function Rem rd(ByVal CardNumber As Integer) As Card
...
...
     E le
     ' added
     If CardToGet.GetDescription()(0) = "F" Then
         NumFiles -= 1
     ElseIf CardToGet.GetDescription()(0) = "P" Then
         NumPicks -= 1
     ElseIf CardToGet.GetDescription()(0) = "K" Then
         NumKeys -= 1
     End If
     'end addition
     Return CardToGet

'code added
Public Sub DisplayStats()
    Dim KeyChance, PickChance, FileChance As Single
    KeyChance = NumKeys / GetNumberOfCards() 00
    PickChance = NumPicks / GetNumber rd() * 100
    FileChance = NumFiles / Get rds() * 100
    Console.WriteLine("Th & KeyChance.ToString("F2") &
    will be a key. Fi echance.ToString("F2") & "% chance t
    PickChan g("F2") & "% chance that it will be a pick
End
'end on
```

Changes to GetCardFromDeck

```
                    Console.WriteLine(Hand.GetCardDisplay())
                    'code added
                    Deck.DisplayStats()
                    'end addition
                    Console.Write("To deal with this you need to either
```

Testing:

- Showing the percentages for at least one tool (even if incorrect) to two decimal places which card you would like to select or whether to discard from the deck.

  Note that the percentages are unlikely to match the ones below. [1 mark] ↓

```
Difficulty encountered!

HAND:

------------------------
| P b | F b | P c | P b |
------------------------

There is a 28.57% chance that the next card will be a key, a 21.43% chan
 it will be a file and a 35.71% chance that it will be a pick.
To deal with this you need to either lose a key (enter 1-5 to specify po
of key) or (D)iscard five cards from the deck:>
```

Task 7

Coding:

- Adding one multi-tool of each kind to the deck at creation time [1 mark]
- Adding one multi-tool of each kind to the deck whenev[illegible] is solved [1 mark]
- Adding the three `SetCardToolkit` methods that [illegible]fully allow a card's toolk[illegible]
- Changing `PlayCardToSequence` to ask [illegible]olkit the player would like whene[illegible]
- Calling `SetCardToolkit` for th[illegible]rrec[illegible]ard and toolkit from `PlayCardToSeque[illegible]`

Example Solu[illegible]

Changes to Cre[illegible]ndardDeck

```
            Next
            'code added
            NewCard = New ToolCard("P", "m")
            Deck.AddCard(NewCard)
            NewCard = New ToolCard("F", "m")
            Deck.AddCard(NewCard)
            NewCard = New ToolCard("K", "m")
            Deck.AddCard(NewCard)
            'end addition
        End Sub
```

Changes to `ProcessLockSolved`

```
            End While
            'code added
            Dim NewCard As Card
            NewCard = New ToolCard("P", "m")
            Deck.AddCard(NewCard)
            NewCard = New ToolC[illegible]("[illegible]", "m")
            Deck.AddCard([illegible]
            NewC[illegible]d [illegible]olCard("K", "m")
            D[illegible]dd[illegible]a(NewCard)
            [illegible]dition
            Deck.Shuffle()
```

Changes to `PlayCardToSequence`

```
        Private Sub PlayCardToSequence(ByVal CardChoice As Integer)
            'code added
            If Hand.GetCardDescriptionAt(CardChoice - 1)(2) = "m" Then
                Dim ToolKit As String
                Console.WriteLine()
                Console.Write("Which toolkit would you like to choose? "
                ToolKit = Console.ReadLine()
                Hand.SetCardToolkit(CardChoice - 1, ToolKit)
            End If
            'end addition
```

Creation of `SetCardToolkit` in `CardCollection`

```
        Public Sub SetCardToolkit(P[illegible]ti[illegible]s Integer, ToolKit As String)
            If Cards(Position).[illegible]etD[illegible]ription()(2) = "m" Then
                Cards(P[illegible]n [illegible]SetCardToolkit(ToolKit)
            [illegible]
        End S[illegible]
```

Creation of `SetCardToolkit` in ToolCard

```
        Public Overrides Sub SetCardToolkit(ToolKit As String)
            Kit = ToolKit
        End Sub
```

Creation of `SetCardToolkit` in Card

```
        Public Overridable Sub SetCardToolkit(ToolKit As String)

        End Sub
```

Testing:

- Showing the sequence updated with the card played of the `toolkit` chosen [1 m

```
SEQUENCE:

-----------------------------
| P b | K b | F b | P a | F a |
-----------------------------


HAND:

-----------------------------
| F c | P c | P m | P b | K c |
-----------------------------


(D)iscard inspect, (U)se card
Enter a number between 1 and 5 to specify card to use:> 3
(D)iscard or (U)se?
Which tool do you like to choose? a

A challenge on the lock has been met.


Current score: 30

CURRENT LOCK
------------
Challenge met: P a, F a, P a
Not met:       K b


SEQUENCE:

-----------------------------
| P b | K b | F b | P a | F a |
-----------------------------


HAND:

-----------------------------
| F c | P c | P b | K c | P a |
-----------------------------
```

Task 8

- Changing PlayGame to pass in the argument for the sequence to GetLockDetai
  to accept the new parameter [1 mark]
- Changing GetLockDetails to match a single card on the [illegible]uence to the first ca
  message to partially met, also to not crash when t[illegible] is an empty sequence [1 m
- Changing GetLockDetails to generate [illegible]lly met message when only the la
  first card of a challenge [1 mark]
- Changing GetLockDet[illegible] to generate a partially met message when the last tw
  first two card[illegible] c[illegible]ge. [1 mark]

Example Soluti[illegible]

Changes to GetLockDetails (Note the lengthy embedded if statement a
subsequent conditions of an if statement clause joined with and e
false). Solutions that crash because of this should be docked a m

```
                Else
                    'change, current condition moved to new else clause
                    Dim Condition As List(Of String) = C.GetCondition()
                    If Condition.Count = 3 Then
                        Dim SeqLen As Integer = Sequence.GetNumberOfCard
                        If SeqLen > 0 Then
                            If Condition(1) = Sequence.GetCardDescriptio
                            Sequence.GetCardDescriptionAt(SeqLen - 1) Th
                                LockDetails &= "Partially met: "
                            ElseIf Condition(0) = Sequence.GetCardDescri
                                LockDetails &= "Partially met: "
                            Else
                                LockDetails &= "Not met:       "
                            End If
                        Elseif SeqLen = 0 Then
                            If Condition(0) = Sequence.GetCardDescriptio
                                LockDetails &= "Partially met: "
                            Else
                                LockDetails &= "Not met:       "
                            End If
                        Else
                            LockDetails &= "Not met:       "
                        End If
                    Else
                        LockDetails &= "Not met:       "
                    End If
                    'end change
                End If
```

Changes to PlayGame

```
                    Console.WriteLine("Current [illegible] " & Score)
                    'change
                    Console.WriteLine([illegible]Lock.GetLockDetails(Seq
                    'end chang[illegible]
                    Con[illegible]Wri[illegible]eline(Sequence.GetCardDisplay())
```

Testing:

- First card played to sequence doesn't generate partially met (if it doesn't match) a

```
CURRENT LOCK
------------
Not met:        P a, F a, P a
Not met:        P c, F c, P c


SEQUENCE:

------
| F b |
------

HAND:

----------------------------
| P a | P c | F b | P b | F b |
----------------------------


(D)iscard inspect, (U)se card:> u
Enter a number between 1 and 5 to specify card to use:> 1
(D)iscard or (P)lay?:> p

Current score: 3

CURRENT LOCK
------------
Partially met: P a, F a, P a
Not met:        P c, F c, P c


SEQUENCE:

-----------
| F b | P a |
-----------

HAND:

----------------------------
| P c | F b | P b | F b | P c |
----------------------------
```

- Last two cards on the sequence matching first two of a challenge generates partially met [1 mark] →

```
HAND:

---------------
| P c | K b | P
---------------


(D)iscard inspe
Enter a number
(D)iscard or (P

Current score:

CURRENT LOCK
------------
Partially met:
Not met:


SEQUENCE:

-----------
| F b | P a |
-----------


HAND:

---------------
| P c | K b | P
---------------


(D)iscard inspe
```

Task 9

Coding:

- Adding a variable for bonusCounter and initialising it to 0 for each new lock [1 mark]
- Adding 1 to the variable each time a card is played or discarded [1 mark]
- Awarding the correct bonus once the lock is solved (including 0 if over 20 cards was left) and resetting it to 0 [1 mark]

Example Solution

```
Public Sub PlayGame()
    Dim MenuChoice As String
    If Locks.Count > 0 Then
        GameOver = False
        CurrentLock = New Lock()
        SetupGame()
        While Not GameOver
            LockSolved = False
            'code added
            Dim BonusCounter As Integer = 0
            'end addition
            While Not LockSolved And Not GameOver
                Console.WriteLine()
                Console.WriteLine("Current score: " & Score)
                Console.WriteLine(CurrentLock.GetLockDetails())
                Console.WriteLine(Sequence.GetCardDisplay())
                Console.WriteLine(Hand.GetCardDisplay())
                MenuChoice = GetChoice()
                Select Case MenuChoice
                    Case "D"
                        Console.WriteLine(Discard.GetCardDisplay
                    Case "P"
                        Dim CardChoice As Integer = GetCardChoice
                        Dim DiscardOrPlay As String = GetDiscard
                        'code added
                        BonusCounter += 1
                        'end addition
                        If DiscardOrPlay = "D" Then
                            MoveCard(Hand, Discard, Hand.GetCard
                            GetCardFromDeck(CardChoice)
                        ElseIf DiscardOrPlay = "P" Then
                            PlayCardToSequence(CardChoice)
                        End If
                End Select
                If CurrentLock.GetLockSolved() Then
                    LockSolved = True
                    ProcessLockSolved()
                    'code added
                    Dim Bonus As Integer = Math.Max(0, 20 - Bonus
                    Score += Bonus
                    Console.WriteLine("The [illegible] awarded you " &
                    BonusCounter = 0
                    'end addition
                End If
            End While
            GameOver = CheckIfPlayerHasLost()
        End While
    Else
        Console.WriteLine("No locks in file.")
    End If
End Sub
```

Testing:

- Solving a lock in under 20 cards and getting the correct bonus (which doesn't have [1 mark] ↓

```
A challenge on the lock has been met.

Lock has been solved.  Your score is now: 31
This lock awarded you 17 bonus points.

Current score: 38

CURRENT LOCK
------------
Not met:        F c, P c, F c



SEQUENCE:

-----------------
| P b | K b | F b |
-----------------


HAND:

-----------------------------
| P a | P a | P b | K c | F c |
-----------------------------


(D)iscard inspect (U)se
```

- Solving a lock in over 20 cards and getting 0 bonus. [1 mark] ↓

```
A challenge on the lock has been met.

Lock has been solved.  Your score is now: 38
This lock awarded you 0 bonus points.

Current score: 38

CURRENT LOCK
------------
Not met:        P a, F a, K a
```

Task 10

Coding:

- Modifying SetupGame to have a 25% chance of adding a GeniusCard [1 mark]
- Modifying ProcessLockSolved to have a 25% chance of adding a GeniusCard
- Creating a GeniusCard class that inherits from Card, has a constructor with self CardType to Gen [1 mark]
- Asking the user to enter a challenge number or discard when a genius card is drawn
- Processing the GeniusCard correctly to solve the challenge chosen [1 mark]
- Processing the GeniusCard correctly to discard it [1 mark]
- Handling the adding of a GeniusCard correctly if drawn while refilling the hand print a message [1 mark]

Example Solution

Creation of AddGeniusCardToDeck

```
Private Sub AddGeniusCardToDeck()
    Deck.AddCard(New GeniusCard())
End Sub
```

Creation of GeniusCard

```
'code added
Class GeniusCard
    Inherits Card

    Protected CardType As String

    Sub New()
        MyBase.New()
        CardType = "Gen"
    End Sub

    Public Sub New(ByVal CardNo As Integer)
        CardType = "Gen"
        CardNumber = CardNo
    End Sub

    Public Overrides Function GetDescription() As String
        Return CardType
    End Function
End Class
'end addition
```

Changes to SetupGame

```
AddDifficultyCardsToDeck()
'code added
If RNoGen.Next(1, 4) = 1 Then
    AddGeniusCardToDeck()
End If
'end addition
Deck.Shuffle()
```

Changes to ProcessLockSolved

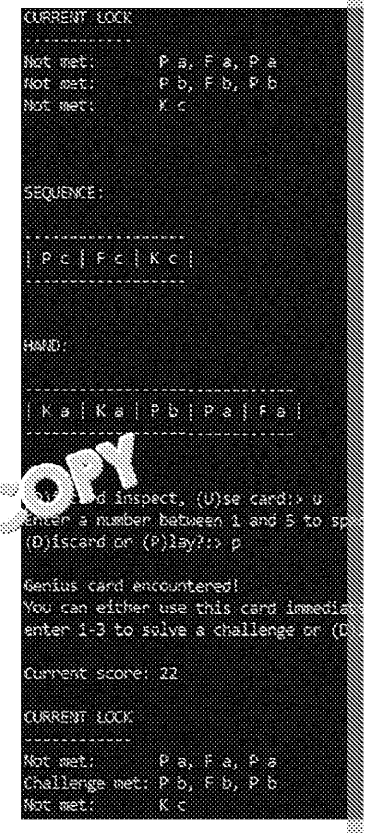
```
End While
'code added
If RNoGen.Next(1, 4) = 1 Then
    AddGeniusCardToDeck()
End If
'end addition
Deck.Shuffle()
```

Changes to GetCardFromDeck

```
                CurrentCard.Process(Deck, Discard, Hand, Sequence, C
                'code added
            ElseIf Deck.GetCardDescriptionAt(0) = "Gen" Then
                Dim CurrentCard As Card = Deck.Re[illegible]eCard(Deck.GetCa
                Console.WriteLine()
                Console.WriteLine("Genius ca[illegible] encountered!")
                Console.WriteLine[illegible]etCardDisplay())
                Console.Wri[illegible]e("[illegible]an either use this card immediat
                disc[illegible]
                [illegible].Write("Enter 1-" & CurrentLock.GetNumberOfCh
                challenge or (D)iscard it so it can come up after re
                Dim Choice As String = Console.ReadLine().ToUpper()
                Console.WriteLine()
                If Choice = "D" Then
                    Discard.AddCard(CurrentCard)
                Else
                    CurrentLock.SetChallengeMet(Integer.Parse(Choice
                End If
                'end addition
            End If
        End If
        While Hand.GetNumberOfCards() < 5 And Deck.GetNumberOfCards(
                If Deck.GetCardDescriptionAt(0) = "Dif" Then
                    MoveCard(Deck, Discard, Deck.GetCardNumberAt(0))
                    Console.WriteLine("A difficulty card was discard
                    the hand.")
                    'code added
                ElseIf Deck.GetCardDescriptionAt(0) = "Gen" Then
                    MoveCard(Deck, Discard, Deck.GetCardNumberAt(0))
                    Console.WriteLine("A [illegible] card was discarded from
                    'end addit[illegible]
                Else
                    [illegible](Deck, Hand, Deck.GetCardNumberAt(0))
```

Testing:

- Using a GeniusCard successfully [1 mark] →



- Discarding a `GeniusCard` successfully [1 mark] ⬇

```
Genius card encountered!
You can either use this card immediately to solve a challenge or discard it.
enter 1-3 to solve a challenge or (D)iscard it so it can come up after reshuff

Current score: 30

CURRENT LOCK
------------
Challenge met: P a, F
Challenge [illegible] B
Not met: [illegible]


SEQUENCE:

---------------------------------
| P c | F c | K c | P a | F a | P a |
---------------------------------


HAND:

-----------------------------
| K a | K b | P b | K c | P b |
-----------------------------


(D)iscard inspect, (U)se card:> u
Enter a number between 1 and 5 to specif[illegible] case: 1
```

Task 11

Coding:

- Adding the `Credits` attribute and initialising it to 10 [1 mark]
- Asking whether the player would like to buy a tool only when they have played or d credits left [1 mark]
- Ensuring that keys are not listed if the [illegible] lower than 3 credits remaining (even [1 mark]
- Adding a ToolCard of the correct type and toolkit to the player's hand in the fifth p
- Removing the ToolCard from the deck at the correct position [1 mark]
- Deducting the correct number of credits for buying a card [1 mark]
- Printing out a list of the correct number of each tool available and not printing tools [1 mark]
- Printing option 10 correctly at the end of the menu, once and once only [1 mark]
- Having an iteration statement to correctly calculate the number of tools of each typ
- Returning a list from `PrintToolsAvailable` that contains the index of a card wit

Example Solution

Adding the `Credits` attribute

```
        Private LockSolved As Boolean
        'code added
        Private Credits As Integer = 10
        'end addition
```

Changes to `GetCardFromDeck`

```
            If Deck.GetNumberOfCards() > 0 Then
                'code added
                Credits >= 2 Then
                    Console.WriteLine()
                    Console.Write("Would you like to buy a tool(y/n)? ")
                    Dim Choice As String = Console.ReadLine().ToLower()
                    If Choice = "y" Then
                        Dim KeysAvailable As Boolean = False
                        If Credits > 2 Then
                            KeysAvailable = True
                        End If
                        Dim ToolList As List(Of Integer) = Deck.PrintToo
                        Dim CardChosen As Integer
                        Console.Write("Which tool would you like to buy?
                        CardChosen = Integer.Parse(Console.ReadLine())
                        If CardChosen <> 10 Then
                            If ToolList(CardChosen - 1) <> -1 Then
                                MoveCard(Deck, Hand, Deck.GetCardNumberA
                                If CardChosen > 6 Th
                                    Credits -= 3
                                Else
                                    Credits -= 2
                                End If
                            End If
                        End If
                    End If
                End If
                'end addition
                If Deck.GetCardDescriptionAt(0) = "Dif" Then
```

New PrintToolsAvailable method in CardCollection

```
Public Function PrintToolsAvailable(KeysAvailable As Boolean) As
    Dim Tools As List(Of String) = New List(Of String) From {"F
    "P c", "K a", "K b", "K c"}
    Dim ToolList As List(Of Integer) = New List(Of Integer) From
    1, -1}
    Dim ToolsAvailable As List(Of Integer) = New List(Of Integer)
    Dim i As Integer
    For i = 0 To GetNumberOfCards() - 1
        If Cards(i).GetDescription() = "F a" Then
            ToolsAvailable(0) += 1
            If ToolList(0) = -1 Then
                ToolList(0) = i
            End If
        ElseIf Cards(i).GetDescription() = "F b" Then
            ToolsAvailable(1) += 1
            If ToolList(1) = -1 Then
                ToolList(1) = i
            End If
        ElseIf Cards(i).GetDescription() = "F c" Then
            ToolsAvailable(2) += 1
            If ToolList(2) = -1 Then
                ToolList(2) = i
            End If
        ElseIf Cards(i).GetDescription() = "P a" Then
            ToolsAvailable(3) += 1
            If ToolList(3) = -1 Then
                ToolList(3) = i
            End If
        ElseIf Cards(i).GetDescription() = "P b" Then
            ToolsAvailable(4) += 1
            If ToolList(4) = -1 Then
                ToolList(4) = i
            End If
        ElseIf Cards(i).GetDescription() = "P c" Then
            ToolsAvailable(5) += 1
            If ToolList(5) = -1 Then
                ToolList(5) = i
            End If
        ElseIf Cards(i).GetDescription() = "K a" And KeysAvailabl
            ToolsAvailable(6) += 1
            If ToolList(6) = -1 Then
                ToolList(6) = i
            End If
        ElseIf Cards(i).GetDescription() = "K b" And KeysAvailabl
            ToolsAvailable(7) += 1
            If ToolList(7) = -1 Then
                ToolList(7) = i
            End If
        ElseIf Cards(i).GetDescription() = "K c" And KeysAvailabl
            ToolsAvailable(8) += 1
            If ToolList(8) = -1 Then
                ToolList(8) = i
            End If
        End If
    Next

    For i = 0 To Tools.Count - 1
        If ToolList(i) > -1 Then
            Console.WriteLine(i + 1 & ". " & Tools(i) & "(" & Too
        End If
    Next

    Console.WriteLine("10. No Tool (buy nothing)")

    Return ToolList
End Function
```

Testing:

- Buying two tools [1 mark] ↓

```
Enter L to load a game from a file, anything else to play a new game:>

Current score: 0

CURRENT LOCK
------------
Not met:        P a, F a, P a
Not met:        K b

SEQUENCE: emp

HAND:

-----------------------------
| P c | K a | P c | P c | K c |
-----------------------------

(D)iscard inspect, (U)se card:> u
Enter a number between 1 and 5 to specify card to use:> 3
(D)iscard or (P)lay?:> d
Would you like to buy a tool (y/n)? y
1. F a (3 available)
2. F b (3 available)
3. F c (3 available)
4. P a (5 available)
5. P b (5 available)
6. P c (2 available)
7. K a (2 available)
8. K b (3 available)
9. K c (2 available)
10. No Tool (buy nothing)
Which tool wo
Current score

CURRENT LOCK
------------
Not met:        P a, F a, P a
Not met:        K b
```

```
SEQUENCE: empty

HAND:

---------------
| P c | K a | P
---------------

(D)iscard inspe
Enter a number
(D)iscard or (P
Would you like
1. F a (3 avail
2. F b (3 avail
3. F c (3 avail
4. P a (4 avail
5. P b (5 avail
6. P c (2 avail
7. K a (2 avail
8. K b (3 avail
9. K c (2 avail
10. No Tool (bu
Which tool woul

Current score:

CURRENT LOCK
------------
Not met:
Not met:

SEQUENCE:

------
| P a |
------

HAND:

---------------
| P c | K a | P
```

- Trying to buy a tool with 2 credits left [1 mark] ↓

```
Would you like to buy a tool (y/n)? y
1. F a (2 available)
2. F b (3 available)
3. F c (1 available)
4. P a (4 available)
5. P b (4 available)
6. P c (3 available)
10. No Tool (buy nothing)
```

Task 12

Coding:

- Selection statement to check whether the challenge just solved was at least three
- Iteration statement to run once for each tool card in the challenge just solved [1 m
- Call to `MoveCard` to move one tool card from the sequence to the discard pile insi

Example Solution

```
    If CurrentLock.ChallengeConditionMet(SequenceAsString) Then
        'code addition
        If SequenceAsString.Length >= 13 Then
            Dim i As Integer
            For i = 1 To SequenceAsString.Length \ 4
                MoveCard(Sequence, Discard, Sequence.GetCardNumberAt(Seq
            Next
        End If
        'end addition
        Return True
```

Testing:

- Solve a challenge with one card, then with three cards [1 mark] ↓

```
CURRENT LOCK
------------
Not met:      P a, F a, P a
Challenge met: K b



SEQUENCE:

-----------------
| K b | P a | F a |
-----------------


HAND:

-----------------------------
| F b | K a | K c | K b | P a |
-----------------------------


(D)iscard inspect, (U)se card:> u
Enter a number between 1 and 5 to specify card to use:> 5
(D)iscard or (P)lay?:> p

A challenge on the lock has been met.

Lock has been solved. Your score is now: 27

Current score: 27

CURRENT LOCK
------------
Not met:      F c, P c, F c


SEQUENCE:

------
| K b |
------
```

Task 13

Coding:

- Changing `GetChoice` to correctly prompt you to (S)ave the game and `PlayGame`
  'S' was chosen [1 mark]
- Returning a string of the correct format for the save file from `GetChallengesAsS`
- Returning a string of the correct format for the save file from `GetChallengesMet`
- Saving the current score to the game file [1 mark]
- Saving the current lock to the save game file [1 mark]
- Saving the hand, sequence, deck and discard pile to the save game file [1 mark]
- Having a method or a loop that creates the string for a `CardCollection` in the co
  [1 mark]

Example Solution

Changes to `GetChoice`

```
            Console.WriteLine()
            'code change
            Console.Write("(D)iscard inspect, (S)save or (U)se card:> ")
            'end change
            Dim Choice As String = Console.ReadLine().ToUpper()
```

Changes to `PlayGame`

```
                            Console.WriteLine(Di[illegible]d.GetCardDisplay
                            'code added
                        Case "S"
                            Sa[illegible]me
                            'end addition
                        Case "U"
```

Code for SaveG

```
        'code added
        Sub SaveGame()
            Dim SaveName As String = "game2.txt"
            Dim DeckStr As String = ""
            Dim DiscardStr As String = ""
            Dim HandStr As String = ""
            Dim SeqStr As String = ""
            Dim i As Integer

            Try
                Using Writer As StreamWriter = New StreamWriter(SaveName
                    Writer.WriteLine(Score)
                    Writer.WriteLine(CurrentLock.GetCha[illegible]lengesAsString()
                    Writer.WriteLine(CurrentLock.[illegible]a.lengesMetAsStrin
                    For i = 0 To Hand.GetNumb[illegible]rO[illegible]us() - 1
                        If HandStr.Leng[illegible] 0 Then
                            Han[illegible]
                        [illegible]
                        [illegible] Str &= Hand.GetCardDescriptionAt(i) & " " &
                    [illegible]
                    Writer.WriteLine(HandStr)
                    For i = 0 To Sequence.GetNumberOfCards() - 1
                        If SeqStr.Length > 0 Then
                            SeqStr &= ","
                        End If
```

```
                SeqStr &= Sequence.GetCardDescriptionAt(i) & " "
            Next
            Writer.WriteLine(SeqStr)
            For i = 0 To Discard.GetNumberOfCards() - 1
                If DiscardStr.Length > 0 Then
                    DiscardStr &= ","
                End If
                DiscardStr &= Discard.GetCardDescriptionAt(i) & " "
            Next
            Writer.WriteLine(DiscardStr)
            For i = 0 To Deck.GetNumberOfCards() - 1
                If DeckStr.Length > 0 Then
                    DeckStr &= ","
                End If
                DeckStr &= Deck.GetCardDescriptionAt(i) & " " & S
            Next
            Writer.WriteLine(DeckStr)
        End Using
        Console.WriteLine("File saved.")
    Catch
        Console.WriteLine("File not saved")
    End Try
End Sub
'end addition
```

Code for GetChallengesAsString

```
Public Function GetChallengesAsString() As String
    Dim ChallengeStr As String = ""
    For Each C In Challenges
        If ChallengeStr.Length > 0 Then
            ChallengeStr &= ";"
        End If
        ChallengeStr &= ConvertConditionToString(C.GetCondition()
    Next
    Return ChallengeStr
End Function
```

Code for GetChallengesMetAsString

```
Public Function GetChallengesMetAsString() As String
    Dim ChallengeStr As String = ""
    For Each C In Challenges
        If ChallengeStr.Length > 0 Then
            ChallengeStr &= ";"
        End If
        If C.GetMet() Then
            ChallengeStr &= "Y"
        Else
            ChallengeStr &= "N"
        End If
    Next
    Return ChallengeStr
End Function
```

Testing:

- Saving game then loading game [1 mark] ↓

```
(D)iscard inspect, (S)ave or (U)se card?:> s
File saved.

Current score: 8

CURRENT LOCK
------------
Not met:       c a
Challenge      b
Not met:       c


SEQUENCE:

------
| K b |
------


HAND:

-----------------------------
| P b | P c | P b | P c | F b |
-----------------------------


(D)iscard inspect, (S)ave or (U)se card?:> 
```

```
Enter L to load a saved file, anything else to play a new game:> l

Current score: 8

CURRENT LOCK
------------
Not met:       K a
Challenge met: K b
Not met:       K c


SEQUENCE:

------
| K b |
------


HAND:

-----------------------------
| P b | P c | P b | P c | F b |
-----------------------------


(D)iscard inspect, (S)ave or (U)se card?:> 
```

Task 14

Coding:

- Adding 5 to the `BonusPool` after adding 5 to the score when completing a challen
- Adding 5 to the `BonusPool` when playing a card to the sequence that is a partial s
- Resetting the `BonusPool` to 0 under all circumstances where a card is not played
- Creating the new attribute `BonusPool` and initialising it to 0 [1 mark]
- Writing the code for `IsPartial` so that it returns True if the card just played adde add to an existing chain and changing `GetCardDescriptionAt` to return an e

Example Solut

Changes to `PlayCardToSequence`

```
                Console.WriteLine()
                'code change
                Score += 5 + BonusPool
                 'end change
                 'code added
                BonusPool += 5
            Else
                If CurrentLock.IsPartial(Sequence) Then
                    BonusPool += 5
                Else
                    BonusPool = 0
                End If
                'end addition
            End If
```

Changes to `PlayGame`

```
                                GetCardFromDeck(CardChoice)
                                'code added
                                BonusPool = 0
                                'end addition
                            ElseIf DiscardOrPlay = "P" Then
```

Code for `IsPartial`

```
        'code added
        Public Function IsPartial(Seq As CardCollection) As Boolean
            Dim Part As Boolean = False
            For Each C In Challenges
                Dim Condition As List(Of String) = C.GetCondition()
                If Condition.Count = 3 Then
                    Dim SeqLen As Integer = Seq.GetNumberOfCards() - 1
                    If SeqLen > 0 And Condition(1) = Seq.GetCardDescript
                    = Seq.GetCardDescriptionAt(SeqLen - 1) Then
                        Part = True
                    ElseIf SeqLen >= 0 And Condition(0) = Seq.GetCardDes
                        Part = True
                    End If
                End If
            Next
            R    P
        End F    n
        'end a    on
```

Changes to GetCardDescriptionAt

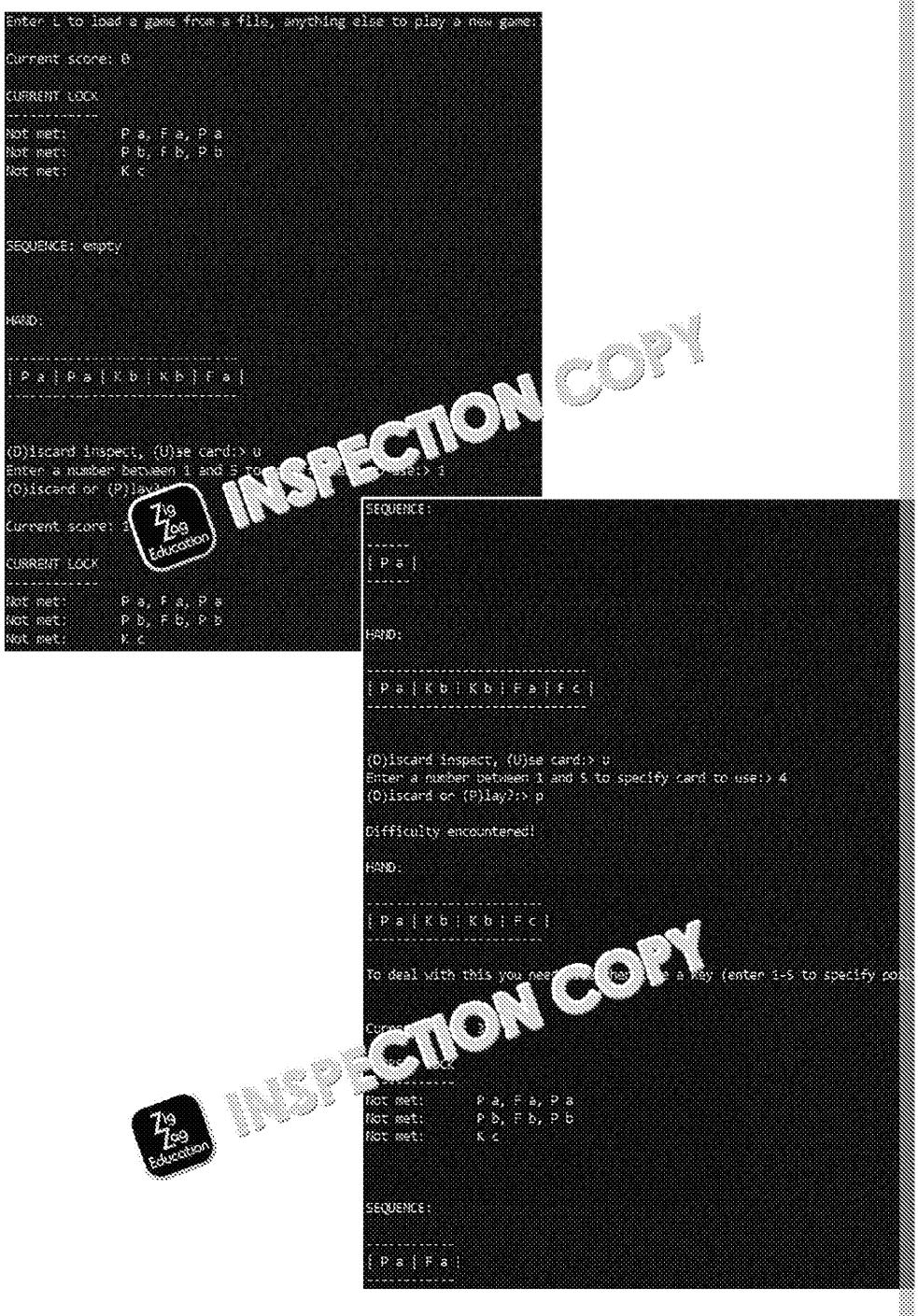
```
Public Function GetCardDescriptionAt(ByVal X As Integer) As Strin
    'code added
    If X < 0 Then
        Return ""
    End If
    'end addition
    Return Cards(X).GetDescriptio
```

Code for new BonusPoolatbate

```
Private Bonus s Integer = 0
```

Testing:

- Playing three cards to solve a challenge then solve it one card after another [1 ma



```
HAND:

-----------------------------
| P a | K b | F c | K c | K a |
-----------------------------


(D)iscard inspect, (U)se card:> u
Enter a number between 1 and 5 to specify card to use
(D)iscard or (P)lay?:> p

A challenge on the lock has been met.


Current score:

CURRENT LOCK
------------
Challenge met: P a, F a, P a
Not met:       P b, F b, P b
Not met:       K c


SEQUENCE:

------------------
| P a | F a | P a |
------------------


HAND:

-----------------------------
| K b | F c | K c | K a | P c |
-----------------------------


(D)iscard inspect, (U)se card:> u
Enter a number between 1 and 5 to specify card to use:> 3
(D)iscard or (P)lay?:> p

A challenge on the lock has been met.


Current score: 42
```

```
CURRENT LOCK
------------
Challenge met: P a,
Not met:       P b,
Challenge met: K c


SEQUENCE:

--------------------
| P a | F a | P a |
--------------------


HAND:

--------------------
| K b | F c | K a |
--------------------


(D)iscard inspect,
Enter a number betw
(D)iscard or (P)lay

Current score: 44

CURRENT LOCK
------------
Challenge met: P a,
Not met:       P b,
Challenge met: K c


SEQUENCE:

--------------------
| P a | F a | P a |
--------------------


HAND:

--------------------
| K b | K a | P c |
--------------------
```

Task 15

Coding:

- Adding `FinalLock` as a private attribute and initialising it to 0 [1 mark]
- Adding the selection conditions for `FinalLock == 1` and `FinalLock == 2` to Ch don't have the correct contents) [1 mark]
- Changing the condition of the selection statement in `GetCardFromDeck` correctly
- Changing `ProcessLockSolved` to have 10 attempts to find a soluble lock [1 mark
- Changing `ProcessLockSolved` to call `GenerateSolubleLock` once 10 attempts
- Changing `ProcessLockSolved` to set `FinalLock` to 1 and skip the main body of
- Writing `GenerateSolubleLock` such that it always generates a soluble lock (rules the same type excluded) [1 mark]
- Writing `GenerateChallenge` such that it generates a possible challenge from the have two tools of the same type consecutively) [1 mark]
- Returning True and False correctly from `IsSoluble` [1 mark]

Example Solution

Addition of `FinalLock` attribute

```
        'code added
        Private FinalLock As Integer = 0
        'end addition
```

Changes to `CheckIfPlayerHasLost`

```
        Private Function CheckIfPlayerHasLost() As Boolean
            'code added
            If FinalLock = 1 Then
                FinalLock = 2
            ElseIf FinalLock = 2 Then
                Console.WriteLine("You have solved the final lock.  Your
                Return True
            'end addition
            'code change
            ElseIf Deck.GetNumberOfCards() = 0 Then
               Console.WriteLine("You have run out of cards in your deck.
                Return True
            End If
            Return False
            'end change
        End Function
```

Changes to `GetCardFromDeck`

```
            End While
            'code change
            If (Deck.GetNumberOfCards() = 0 And Hand.GetNumberOfCards()
            Hand.GetNumberOfCards() = 0 Then
                'end change'
                GameOver = True
```

Changes to `ProcessLockSolved`

```
        Private Sub ProcessLockSolved()
            Score += 10
            Console.WriteLine("Lock has been solved.  Your score is now:
            'code change
            If FinalLock < 2 Then
                While Discard.GetNumberOfCards() > 0
                    MoveCard(Discard, Deck, Discard.GetCardNumberAt(0))
```

```
            End While
            Deck.Shuffle()
            Dim Attempts As Integer = 0
            While Attempts < 10
                CurrentLock = GetRandomLock()
                If CurrentLock.IsSoluble(Deck, Hand) Then
                    Exit While
                Else
                    Attempts += 1
                End If
            End While
            If Attempts = 10 Then
                Console.WriteLine("Final Lock")
                CurrentLock = GenerateSolubleLock()
                FinalLock = 1
                GameOver = True
            End If
        End If
        'end change
    End Sub
```

Code for GenerateSolubleLock

```
    Private Function GenerateSolubleLock() As Lock
        Dim CardsLeft As List(Of String) = New List(Of String)
        Dim NewLock As Lock = New Lock()
        Dim i As Integer

        For i = 0 To Deck.GetNumberOfCards() - 1
            CardsLeft.Add(Deck.GetCardDescriptionAt(i))
        Next
        For i = 0 To Hand.GetNumberOfCards() - 1
            CardsLeft.Add(Hand.GetCardDescriptionAt(i))
        Next

        NewLock.AddChallenge(GenerateChallenge(CardsLeft))
        NewLock.AddChallenge(GenerateChallenge(CardsLeft))

        Return NewLock
    End Function
```

Code for GenerateChallenge

```
    Private Function GenerateChallenge(ByRef Cards As List(Of String))
        Dim Challng As List(Of String) = New List(Of String)
        Dim CardToRemove As Integer
        Dim i As Integer

        Try
            CardToRemove = RNoGen.Next(0, Cards.Count - 1)
            Challng.Add(Cards(CardToRemove))
            Cards.RemoveAt(CardToRemove)
            For i = 0 To RNoGen.Next(0, 2)
                CardToRemove = RNoGen.Next(0, Cards.Count - 1)
                While Cards(CardToRemove)(0) = Challng(Challng.Count
                    CardToRemove = RNoGen.Next(0, Cards.Count - 1)
                End While
                Challng.Add(Cards(CardToRemove))
                Cards.RemoveAt(CardToRemove)
            Next
        Catch ex As Exception
            'ran out of cards so we go with what we have so far
        End Try

        Return Challng
    End Function
```

Code for IsSoluble

```
Public Function IsSoluble(Deck As CardCollection, Hand As CardCo
    Dim CardsLeft As List(Of String) = New List(Of String)
    Dim ChallengesLeft As List(Of String) = New List(Of String)
    Dim i As Integer

    For i = 0 To Deck.GetNumberOfCard ()
        CardsLeft.Add(Deck.Get  scriptionAt(i))
    Next
    For i = 0 To H  mberOfCards() - 1
        Card  d(Hand.GetCardDescriptionAt(i))
    
    F  C In Challenges
        For Each S In C.GetCondition()
            ChallengesLeft.Add(S)
        Next
    Next
    For Each S In ChallengesLeft
        If Not CardsLeft.Remove(S) Then
            Return False
        End If
    Next

    Return True
End Function
```

Testing:

- Printing out the final lock [1 mark] ➔

```
HAND:

---------------------
| P b | K a | P b | P
---------------------


(D)iscard inspect, (U)
er a number between
(D)iscard or (P)lay?:>

Current score: 74

CURRENT LOCK
------------
Challenge met: P a, F
Not met:       P c, K


SEQUENCE:

---------------------
| P a | K c | P c | F
---------------------
| P a | F c | P c | K
---------------------


HAND:

---------------------
  b | P b | P a | K
---------------------


(D)iscard inspect, (U)
Enter a number between
(D)iscard or (P)lay?:>

A challenge on the loc

Lock has been solved.
You have solved the fi
```

Name	

ZigZag Education supporting

A Level AQA Computer Science Paper 1

Summer 2022

BREAKTHROUGH!

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:

Programming Theory Question

Answer all questions. Remember to save this document

Q		Answer
1	(a)	
	(b)	
2	(a)	
	(b)	
	(c)	
3	(a)	
	(b)	
4	(a)	
	(b)	
5	(a)	
	(b)	
6	(a)	
	(b)	

(a)

Count	SequenceAsString	Return value
	""	
5		

(b)

Count	SequenceAsString	Return value
	""	
5		

Q		Answer
7	(a)	
	(b)	
8	(a)	
	(b)	
9	(a)	
	(b)	
	(c)	
10	(a)	
	(b)	
11		
12		
13	(a)	
	(b)	
14	(a)	
	(b)	
	(c)	
15		

Programming Tasks

Answer all questions. Remember to save this document

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