Programming Practice Booklet - Collyers Sixth Form College

Evidence Booklet for the Computer Science Programming Exercises

Name: Write your name here. **Block:** Write your class here.

Contents

1: HelloWorld 4

2: Favourites 5

3: Quotation 6

4: WelcomeUser 7

5: FindSumAverage 8

6: FindProduct 9

7: GiveChange 10

8: CalculateTip 11

9: HolidaySpending 12

10: GcseAvg 13

11: CalculateTax 14

12: FtoC 15

13: CalculateFuel 16

14: Liverchester 17

15: AgeSelection 18

16: AgeSelection2 19

17: AqaCalc 20

18: WhileLoop 21

19: Ducky (While Loop with Break) 22

20: CalcAvg 23

21: NeverEndingLoop 24

22: NumberGuess 25

23: ForLoop 26

24: topStudent 27

25: TimesTable 28

26: StepFive 29

27: Journey 30

28: EvensAndOdds 31

29: PiratesGold 32

30: MoneyDispenser 33

31: MinibusDriver 34

32: NameOrder 35

33: Discounts..37
34: Validation..38
35: PasswordEntry..39
36: MaxAndMin..40
37: MultiplicationTable..41
38: Factors..42
39: ArrayNames..44
40: ArrayMarks..45
41: StorageCases..47
42: MessageEncryption..48
43: TallyArray..50
44: LinearSearch..51
45: FindSumAverage2..52
46: Factorial..53
47: SentenceAnalysis..54
48: Scramble..56
49: ConstantVat..58
50: CathedralTowns..59
51: Bubble Sort..61
52: WriteToFile..62
53: ReadFromFile..63
54: WriteBinary..64
55: ReadBinary..65
56: Fibonacci Sequence..66
57: ExceptionHandling..68
58: DateValidation..69
59: PaceCalculator..70
61 Word Games..71
62 Prime Numbers..72
63 Run Length Encryption..73

1: HelloWorld

Task

Write a function to display the message “Hello World!”

Code

Screenshot

2: FAVOURITES

Task

Write a function to display your five favourite bands (or food items) on separate lines.

Code

Screenshot

3: QUOTATION

Task

Write a function to display your favourite quotation. Display the name of the quotation's author on a separate line.

Code

Screenshot

4: WELCOMEUSER

Task

Write a function to read in the user's name and then display a message to welcome them by name (e.g. "Hello Fred!"

Code

Screenshot

5:FINDSUMAVERAGE

Task

Write a function to read in two numbers and print their sum and average. Run your function to show that it works with the numbers 12 and 15.

Code

Screenshots

6:FINDPRODUCT

Task

Write a function to read in two numbers and print their product. Run your function to show that it works with the numbers 12 and 15.

Code

Screenshots

7: GIVECHANGE

Task

Write a function to read in the amount of the bill and the amount of money tendered. Calculate and display the amount of change due to the customer. Run your function to show that it works with a bill amount of £8.50 and an amount tendered of £10.

Code

Screenshots

8: CalculateTip

Task

Write a function to read in the amount of a restaurant bill and display the amount of tip to be left if the percentage tip is 10% or 12.5%. Run your function to show the amounts on a bill of £87.

Code

Screenshots

9:HOLIDAYSPENDING

Task

Write a function to handle holiday spending by carrying out the following steps:

- Set a variable to store the initial amount of holiday spending money (50,000)
- Output the title ‘Italian Holiday’
- Output the number of euros at the start of the holiday
- Read in the number of euros spent on food
- Read in the number of euros spent on trips
- Read in the number of euros spent on presents

Output the amount of money left

Code

Screenshots

10: GCSEAVG

Task

Write a function to read in the total GCSE points score and the number of GCSE's taken. Display the average GCSE score. Show the program working with a total score of 61 and number of GCSEs of 11.

Code

Screenshot

11:CALCULATETAX

Task

Write a function to read in the amount of gross pay that the employee has earned. Calculate and display the amount of tax due (at 25%) and the amount of net pay (gross – tax). Run your function to show that it works with a gross pay figure of £375.

Code

Screenshots

12: FtoC

Task

Temperatures can be converted from Fahrenheit to Centigrade using the following formula, where F is the temperature in Fahrenheit and C the temperature in Centigrade:

C = 5 (F-32) / 9

Write a function to input a Fahrenheit temperature and output the equivalent temperature in Centigrade.

Extension: modify the output to display the temperature to two decimal places.

Code

Screenshots

13:CalculateFuel

Task

Write a function to read in the amount of miles travelled and the amount in **litres** of fuel used. Calculate and display the fuel economy in miles per **gallon**. Run your function to show that it works with inputs of 300 miles and 40 litres of fuel.

Code

Screenshots

14: LIVERCHESTER

Task

Liverchester FC Supporters Club organises coaches to take members to away matches. Between 8am and 9am members turn up at the home ground where there is a line of coaches. They always board the front coach. When a coach is full, it sets off. At 9am the current front coach sets off, whether or not it is full. Write a function to carry out the following tasks:

- Input the total number of club members travelling to a match
- Input the seating capacity of a coach (assume they are all the same)
- Output the number of full coaches
- Output the number of passengers on the last coach

Code

Screenshots

15:AGESELECTION

Task

Write a function that asks the user to input their age. If the age is 15 or less, display the message "You are entitled to purchase a child's ticket". Otherwise, display "You must buy and adult ticket!" Show that the program works correctly with the inputs of 14, 15 and 16.

Code

Screenshots

16:AGESELECTION2

Task

Write a function that asks the user to input their age. Depending on their age, output the relevant message. Screenshot the program working with an input age of 23.

1-15	“You are too young to drive”
16	“You can drive from this year”
17-65	“You are old enough to drive”
66-75	“You need to renew your licence periodically”
Over 75	“You need to renew your licence annually”

Code

Screenshots

17:AqaCalc

Task

For your AS-Level Computing, you will sit two exams:

COMP1 – 60% of AS (2 hour on-screen exam) 100 marks

COMP2 – 40% of AS (1 hour written exam) 60 marks

Write a function to read in the marks achieved for each paper and output the total score, percentage and grade (>=80% A, 70-79% B, 60-69% C, 50-59% D, 40-49% E, <40% U)

Extension task:

Make it work for the A-level grade (COMP1 and COMP2 are 30% and 20% of A2 respectvely)

COMP3 – 30% of A2 (2 hour 30 minutes written exam) 100 marks

COMP4 – 20% of A2 (The Computing Project) 75 marks

Code

Screenshots

18:WHILELOOP

Task

Write a function that uses a “while loop” to display on the screen the numbers 1 to 9.

Code

Screenshots

19:Ducky (While Loop with Break)

Task

Write a function that uses a “while loop” to ask the user to type in ‘ducky’ and loop endlessly until user types it in. Show it working with the words: **cup, teapot, cake, ducky** entered in sequence.

Code

Screenshots

20: CALCAVG

Task

Write a function that uses a “while loop” to ask the user to type in numbers until they enter the number 999. Display the number of numbers entered, their sum and mean (excluding the final 999). Demonstrate it working with the number sequence: 15,23,4,99,12,57,21,999.

Screenshots

21:NeverEndingLoop

Task

Write a function that uses an endless “while loop” that displays the numbers from 5 onwards.

Code

Screenshots

22:NumberGuess

Task

Write a function that generates a random number between 1 and 100 and then asks the user to guess it. Allow them 5 guesses before the game is “lost”. Help them by displaying whether the guess is too high or too low after each attempt. Make sure that if they guess correctly that they WIN!

Code

Screenshots

23:ForLoop

Task

Write a function that uses a “for loop” to count from 1 to 10

Display as: Number 1, Number 2 etc…

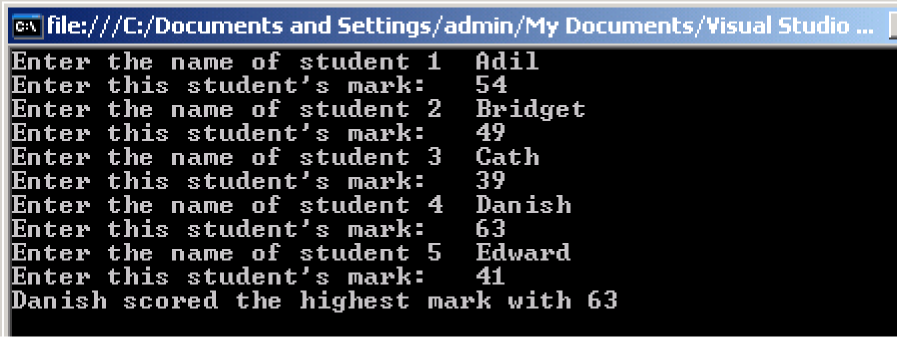
Code

Screenshots

24:TOPSTUDENT

Task

Write a function that inputs the names and marks of five students. The function should then output a statement saying who has the highest mark and what that mark is.



file:///C:/Documents and Settings/admin/My Documents/Visual Studio ...

```
Enter the name of student 1  Adil
Enter this student's mark:   54
Enter the name of student 2  Bridget
Enter this student's mark:   49
Enter the name of student 3  Cath
Enter this student's mark:   39
Enter the name of student 4  Danish
Enter this student's mark:   63
Enter the name of student 5  Edward
Enter this student's mark:   41
Danish scored the highest mark with 63
```

Think carefully about what variables you will need. You will need variables to store data as it is entered, but you will need others to keep a record of the best student to date. Think also about the type of loop you should use.

Code

Screenshots

25:TIMESTABLE

Task

Write a function that asks the user for a number, then display the times table for that number.

E.g. user types in “2”, The function should display:

1 * 2 = 2

2 * 2 = 4

3 * 2 = 6 Etc…

You should use a “for loop” tominimise the lines of code. Demonstrate your program working with the three times table.

Code

Screenshots

26:StepFive

Task

Write a function that makes use of “for loop” to display all of the multiples of 5 between 1 and 100.

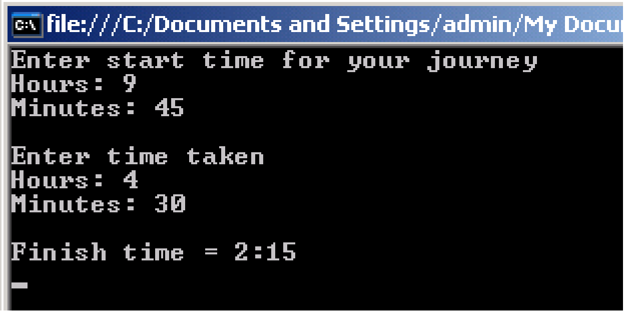
Code

Screenshots

27: JOURNEY

Task

Write a function to help with journey planning. The function should input the starting time and the time taken, both in hours and minutes. It should output the finishing time (using a 12 hour clock). The hours and minutes should be entered separately as in the console session below. Hint: Use six variables, two for each time (one variable for the hours, one for the minutes). Use integer division and modulus operations.



```
Enter start time for your journey
Hours: 9
Minutes: 45

Enter time taken
Hours: 4
Minutes: 30

Finish time = 2:15
_
```

Code

Screenshots

28:EVENSANDODDS

Task

Write a function that asks the user to enter a whole number then outputs a message to say whether it is even or odd.

Hint: Use a modulus operator to determine whether the number is even or odd.

Code

Screenshots

29:PIRATESGOLD

Task

Write a function that asks how much gold the captain has and how many pirates he wants to share it between. Display how much each pirate will get and how much the captain will be left with.

Show your program working with a pot of 23 gold coins and 4 pirates.

Hint: Use integer division and the modulus operator.

Code

Screenshots

30:MONEYDISPENSER

Task

Write a function that reads in a large amount of money (as a whole number) and calculates and displays how many £20, £10, £5 notes and £2 or £1 coins will be given to dispense this amount. For example £37 would result in 1 x £20, 1 x £10, 1 x£5 and 1 x £2.

Hint: Use integer division and the modulus operator

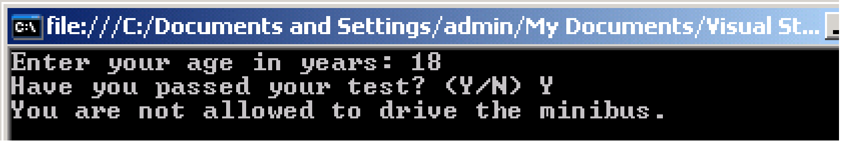
Code

Screenshots

31:MINIBUSDRIVER

Task

A student climbing club has a minibus that can only be driven by drivers over 21 who have passed their test. Write a function that asks the user for their age and whether they have passed their driving test. Then display a message to say whether they can drive the minibus.


file:///C:/Documents and Settings/admin/My Documents/Visual St...

```
Enter your age in years: 18
Have you passed your test? (Y/N) Y
You are not allowed to drive the minibus.
```

Code

Screenshots

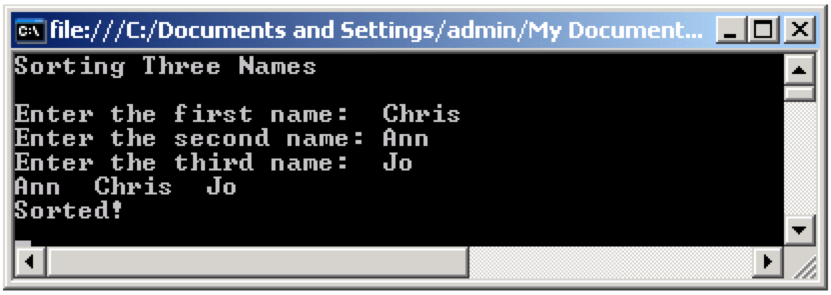
32:NameOrder

Task

Strings of characters can be compared alphabetically using boolean operators in exactly the same way as numbers are compared. For example, the following pseudocode instructions would display the names in alphabetical order:

```
if Name1 < Name2
   then output Name1, Name2
   else output Name2, Name1
endif
```

Write a function that asks for three names and then uses nested IF statements to display them in alphabetical order.



```
Sorting Three Names

Enter the first name:  Chris
Enter the second name: Ann
Enter the third name:  Jo
Ann  Chris  Jo
Sorted!
```

Note 1: You will need to test your function very thoroughly. There are six different ways of arranging any three given names. Your testing should cover all six.

Note 2: You will get unexpected results if you do not use capital letters consistently. This is because of the way that characters are encoded using numeric codes.

Code

Screenshots

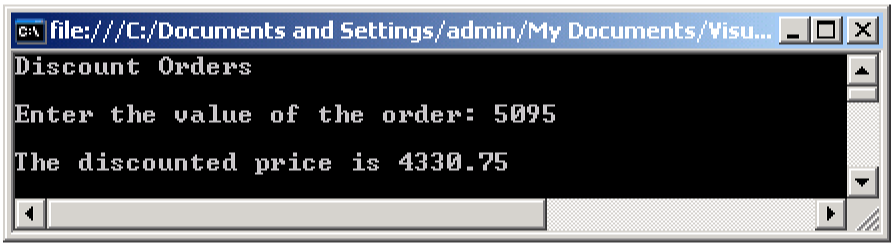
33:DISCOUNTS

Task

An electronics company gives its customers the following discounts:

Order value	< £1000	< £2500	< £5000	< £10,000	£10,000 or more
Discount	0	5%	10%	15%	20%

Write a function to calculate and display the discounted price for orders, as in the following console session:


file:///C:/Documents and Settings/admin/My Documents/Visu...

```
Discount Orders

Enter the value of the order: 5095

The discounted price is 4330.75
```

Code

Screenshots

34:VALIDATION

Task

Write a function to prompt the user to enter a number between 1 and 100. If the number is outside of this range, display the message - **the number entered is not in the valid range** – and ask them to enter another number. Use a loop to keep prompting them until a valid number is entered. When a valid number is entered display the message – **you have entered the number *n*** (where *n* is the number they entered). Display the program working with an input run of 103, -12 and 55

Code

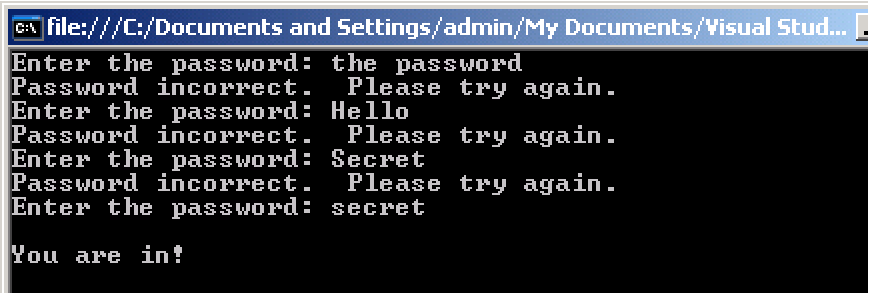
Screenshots

35:PasswordEntry

Task

Write a function that asks the user for a password and keeps asking for it until the word 'secret' is entered.

When the correct password has been entered the message 'You are in!' should be displayed.



file:///C:/Documents and Settings/admin/My Documents/Visual Stud...

```
Enter the password: the password
Password incorrect.  Please try again.
Enter the password: Hello
Password incorrect.  Please try again.
Enter the password: Secret
Password incorrect.  Please try again.
Enter the password: secret

You are in!
```

Code

Screenshots

36: MaxAndMin

Task

Write a function that inputs a succession of numbers until the rogue value '-1' is entered. The function should then output the highest and lowest of the numbers. Think carefully about what kind of loop to use. One loop should be sufficient to enter the values and identify the highest and lowest.

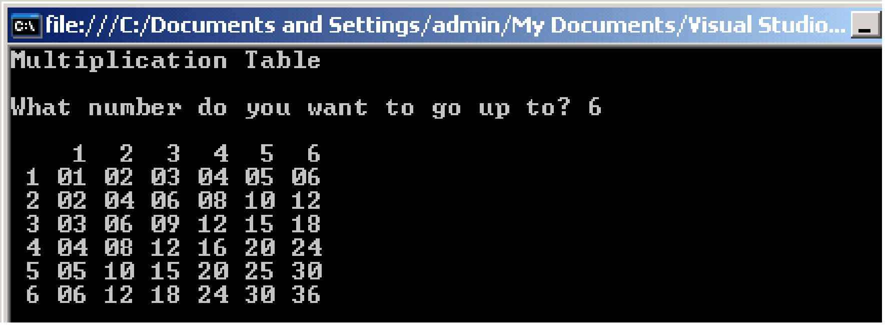
Code

Screenshots

37:MULTIPLICATIONTABLE

Task

Write a function to output a multiplication table in which each entry is the product of the numbers in the top row and left-hand column. The user should be able to choose the size of table.



file:///C:/Documents and Settings/admin/My Documents/Visual Studio...

```
Multiplication Table

What number do you want to go up to? 6

     1  2  3  4  5  6
 1 01 02 03 04 05 06
 2 02 04 06 08 10 12
 3 03 06 09 12 15 18
 4 04 08 12 16 20 24
 5 05 10 15 20 25 30
 6 06 12 18 24 30 36
```

Hint: Use a pair of nested loops for the main part of the table. Try to format the table as neatly as possible.

Code

Screenshots

38: FACTORS

Task

The pseudocode function below should input a positive integer and generate a list of all its factors (including 1, but not including the number itself). It should then output the total number of factors.

Example: Input 6 Output 1

2

3

Number of factors = 3

The function contains at least nine errors. How many can you find?

Note: The line numbers are for reference purposes only; they are not part of the function.

You can assume that all variables have been declared correctly.

```
1    repeat
2       output "Enter a positive integer
3       input TestNumber
4    until TestNumber < 0
5    Count ¬ 1
6    Divisor ¬ 1
7    while TestNumber < Divisor
8       if Divisor mod TestNumber = 0
9                then output Divider
10      endif
11      Count ¬ Count + 1
12      Divisor +1 ¬ Divisor
13   endwhile
14   output "Number of factors = ", Court
```

Convert the pseudocode into working code and demonstrate it working for the number 6.

Code

Screenshots

39: ArrayNames

Task

Write a function that initialises a list (array) with four different names.

Use a for loop to display all of the names.

Extension task:

Modify your function to take in four names from the user and display them on the screen.

Code

Screenshots

40: ArrayMarks

Task

Write a function to initialise a list (array) of eight elements to store the student names shown below. Initialise a second list (array) of eight elements with the marks shown below.

students		marks	
Fred	0	**67**	0
Jack	1	**25**	1
Chris	2	**92**	2
Ali	3	**49**	3
Harry	4	**38**	4
Bill	5	**72**	5
Zak	6	**99**	6
Phil	7	**54**	7

Use a “for loop” to display each name alongside the relevant marks.

Extension task1: extend your function to calculate the average of the number of marks of all the students. Display each student’s name and their marks on the console with a message stating whether the mark is below average, above or equal to average.

Extension task2: extend your function to count how many students were above average and how many below average. Display your results.

Code

Screenshots

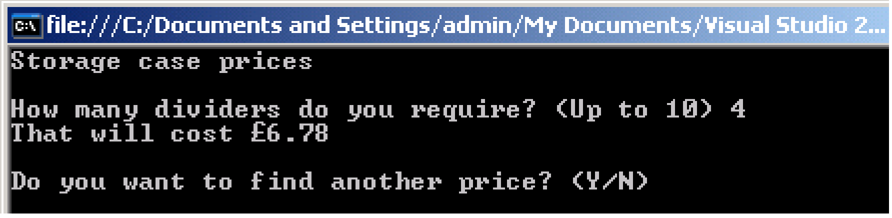
41:StorageCases

Task

An online business sells CD storage cases. The price of each case is £3.50 + 82p for every divider supplied with it. The maximum number of dividers per case is ten.

Write a function that uses a list (array) to store 11 prices. It must then populate the list to contain the correct price for each option by calculating the cost of case + 0 dividers, case + 1 divider, and so on, up to the maximum of case + 10 dividers. You should use a “for loop” to do this.

The function should then ask the user how many dividers are required, look up the corresponding price in the array and state it, as in the console session below. This dialogue should be repeated until the user decides to stop.



file:///C:/Documents and Settings/admin/My Documents/Visual Studio 2...

```
Storage case prices

How many dividers do you require? (Up to 10) 4
That will cost £6.78

Do you want to find another price? (Y/N)
```

Code

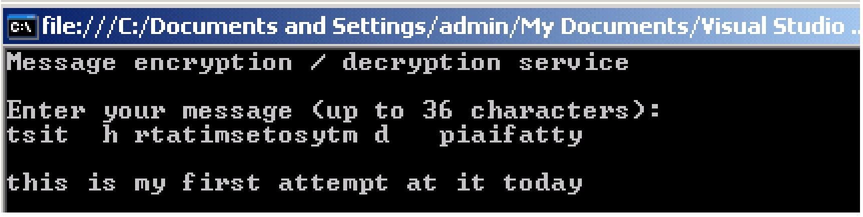
Screenshots

42: MessageEncryption

Task

One simple way to encrypt a short message is to write it letter by letter into a two-dimensional array. The letters are entered row by row, but read out column by column to give a jumbled version. The message can be decrypted by repeating the process.

Write a function that declares an array of characters with six rows and six columns and fills it with 'space' characters. It should then input a message, place it letter by letter into the array by rows, then output the result of reading it column by column. The console session below shows the result of entering a previously scrambled message.


file:///C:/Documents and Settings/admin/My Documents/Visual Studio ..

```
Message encryption / decryption service

Enter your message (up to 36 characters):
tsit  h rtatimsetosytm d   piaifatty

this is my first attempt at it today
```

When testing your function try to use messages that are close to the maximum length or your encrypted version will contain many blanks. When copying down the encrypted version, count the spaces carefully.

Can you think of a way to make your function cope with a wider range of message lengths?

Code

Screenshots

43:TALLYARRAY

Task

Use a "for loop" to simulate the throwing of a dice 100 times. Use the function **randint** from the **random** module to generate the number each time.

Use a six element list (array) to keep a tally of the number of times that each dice number was thrown. At the end of 100 "throws", display the results.

Extension:

Display the value of each element of ***Tally*** as symbols that make it look more like a conventional tally chart.

Code

Screenshots

44: LINEARSEARCH

Task

Write a function to initialise an array with five names. Prompt the user for a name to look for. If the name is found the function should display the index of this item. If the name is not found, a message informing the user should be presented.

Code

Screenshots

45: FindSumAverage2

Task

Open your “FindSumAverage” function (task 4)

Modify your program so that it is split up into the following functions:

- InputData()
- CalculateResults()
- OutputResults()
- Main()

Code

Screenshots

Screenshots

46: Factorial

Task

Write a program, to take a number from the user and display its factorial.

Factorial of 2 = 2 x 1 = 2

Factorial of 3 = 3 x 2 x 1 = 6

Factorial of 4 = 4 x 3 x 2 x 1 = 24

Etc…

The program should comprise three functions:

Main – the main function that calls Getnumber and Factorial and displays the result.

Getnumber - function that gets and **returns** the number from the user

Factorial - function that is passed a number (previously entered) as a parameter and **returns** one value which is the factorial of this number.

Code

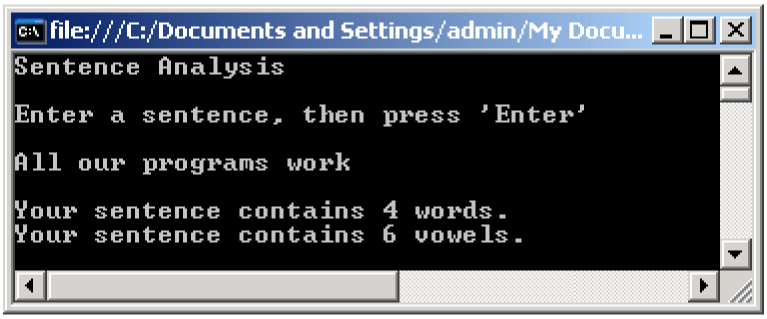
Screenshots

47:SENTENCEANALYSIS

Task

For this exercise you will need to use some of Python's built in string-handling functions. These include functions to return the length of a string and to return a particular character from the middle of a string.

Write a function to analyse a sentence as in the console session shown below.



```
Sentence Analysis

Enter a sentence, then press 'Enter'

All our programs work

Your sentence contains 4 words.
Your sentence contains 6 vowels.
```

You should assume that all words are separated by single spaces and there are no leading or trailing spaces.

Use the following pseudocode as the basis for your main function routine.

```
output "Sentence Analysis"
leave blank line
output "Enter a sentence, then press 'Enter'"
leave blank line
input Sentence
leave blank line
call procedure WordCount(Sentence)
call procedure VowelCount(Sentence)
```

You will need to write two functions:

Function *WordCount* takes a string parameter and outputs to the screen the number of words that the string contains.

Hint: think of the link between the number of spaces and the number of words.

Function *VowelCount* takes a string parameter and outputs to the screen the number of vowels that the string contains. Note that there may be capital letters.

Test your program with several different messages. At least one message should include some vowels that are in capitals.

If you have time, consider how you might adapt your function to cope with leading or trailing spaces. You might also try adding further functions to your program. Examples could be functions to count consonants or word lengths.

Code

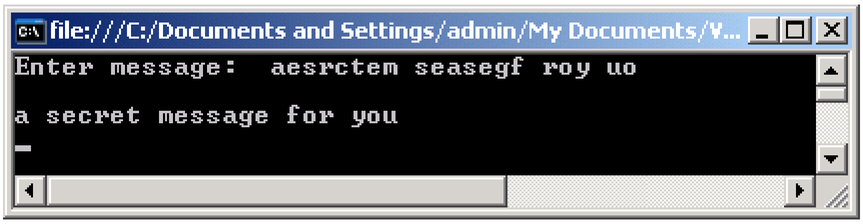
Screenshots

48: SCRAMBLE

Task

Confidential data stored in computer systems is often encrypted (scrambled) to make it difficult for unauthorised persons to decipher. There are many types of encryption. One type is a transposition code in which all the original characters of the message are present, but in a different order.

One of the simplest transposition codes involves swapping adjacent pairs of letters (including any spaces). Thus ‘my secret’ becomes ‘yms ceert’. If this transposition is applied to a message twice, the original message is restored. This means that the same routine can be used both to encrypt and to decrypt messages.



```
Enter message:  aesrctem seasegf roy uo

a secret message for you
_
```

Write a function Scramble that takes a message in the form of a string, swaps adjacent pairs of characters and returns the resulting string to the calling routine. Use this function in a program that asks for a message from the user and then displays it in encrypted form. Note that if there are an odd number of characters the last character remains unchanged at the end.

Test your function by entering a short message, writing down the encrypted form and then re-running the function using the encrypted message as input. Make sure you try messages with an even number of characters and messages with an odd number of characters.

The above method of encryption is not very secure. Can you devise a more elaborate transposition technique that will be harder to crack?

Code

Screenshots

49: CONSTANTVAT

Task

Write a function to take a price of an item from the user

- Calculate the VAT
- Display the VAT amount based on 15%
- Display the price excluding VAT
- Display the price including VAT

Make use of CONSTANTS

Code

Screenshots

50: CATHEDRALTOWNS

Task

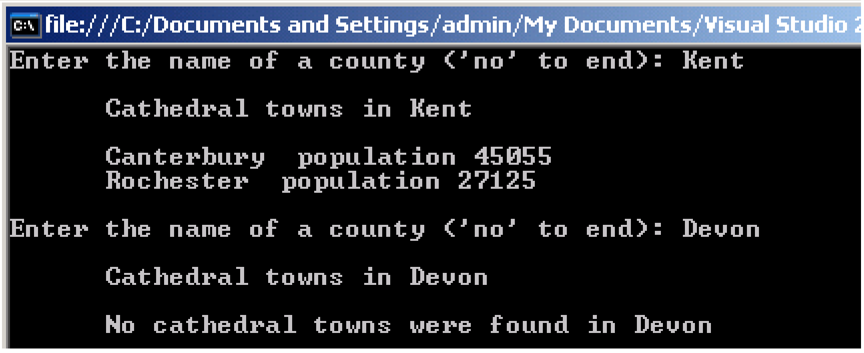
For this function you need to declare a record structure to hold the details of towns. Each record should have three fields:

- The name of the town
- The population
- The county in which the town is located

Create a list (array) to store these records.

The first part of the function should ask for details of an English cathedral town and place them in the array. It should keep doing this until the user decides to stop by typing in a suitable dummy name.

The function should then ask for the name of a county. It should search the array and list any towns belonging to that county together with their population. If there are none, a suitable message should be displayed, as in the dialogue below:



```
file:///C:/Documents and Settings/admin/My Documents/Visual Studio 2
Enter the name of a county ('no' to end): Kent

      Cathedral towns in Kent

      Canterbury  population 45055
      Rochester  population 27125

Enter the name of a county ('no' to end): Devon

      Cathedral towns in Devon

      No cathedral towns were found in Devon
```

This dialogue should be repeated until the user decides to stop.

Use the following table of data to help test your function. You can add data of your own if you wish.

Town	Population	County
Canterbury	45,055	Kent
Lichfield	28,435	Staffordshire
Rochester	27,125	Kent
Truro	20,920	Cornwall

Ripon	16,468	North Yorkshire
Wells	10,406	Somerset

Code

Screenshots

51: Bubble Sort

Task

Write a program that prompts the user to enter 10 names and store them in a list (array). Use the bubble sort algorithm to sort the list and then display the names in alphabetical order. Make use of functions to structure your program.

Code

Screenshots

52: WriteToFile

Task

Write a function that creates a file called students.txt and prompts the user to enter the names of 10 students (use a for loop) and writes each to the file. Copy your code and screenshot the contents of the text file.

Code

Screenshots

53: READFROMFILE

Task

Write a function that reads the file written in the last task and displays each record in turn. Display the message “end of file” when there are no more records to display.

Code

Screenshots

54: WriteBinary

Task

Create a record structure to store the details of a book with the fields: title, ISBN, price and year of publication. Prompt the user to enter the details of several books and write each to a binary file books.bin

Code

Screenshots

55: ReadBinary

Task

Write a function that reads the binary file created in the last task and outputs the titles of each of the books .

Code

Screenshots

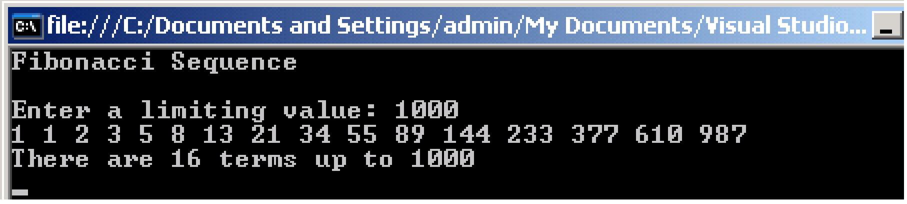
56:FIBONACCI SEQUENCE

Task

The Fibonacci sequence of numbers is named after the medieval Italian mathematician Leonardo Fibonacci, or Leonardo of Pisa. It is a pattern often found in nature, for example in the numbers of petals on many flowers.

Starting with two 1s, each number in the sequence is the sum of the two numbers before it, so it goes 1, 1, 2, 3, 5, 8, 13, 21, and so on.

Write a function to input a limiting value, and then find how many terms there are in the Fibonacci sequence up to that limit. Your function should also output the actual sequence of numbers, as in the following example:


file:///C:/Documents and Settings/admin/My Documents/Visual Studio...

```
Fibonacci Sequence

Enter a limiting value: 1000
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
There are 16 terms up to 1000
_
```

Note: You will need variables to fulfil several different roles here. For example, you will need a 'fixed value' variable to hold the limiting value. As you work through the sequence you will need a variable to hold the most recently calculated number in the sequence and a 'follower' variable to hold the preceding number. You will also need a variable to act as a temporary store when moving terms on one place. Finally, you will need a 'stepper' variable to count the number of terms generated.

Code

Screenshots

57:EXCEPTIONHANDLING

Task

Write a function to prompt the user for an integer and displays a meaningful error message when the user violates this request. You need to use Try...Except for this.

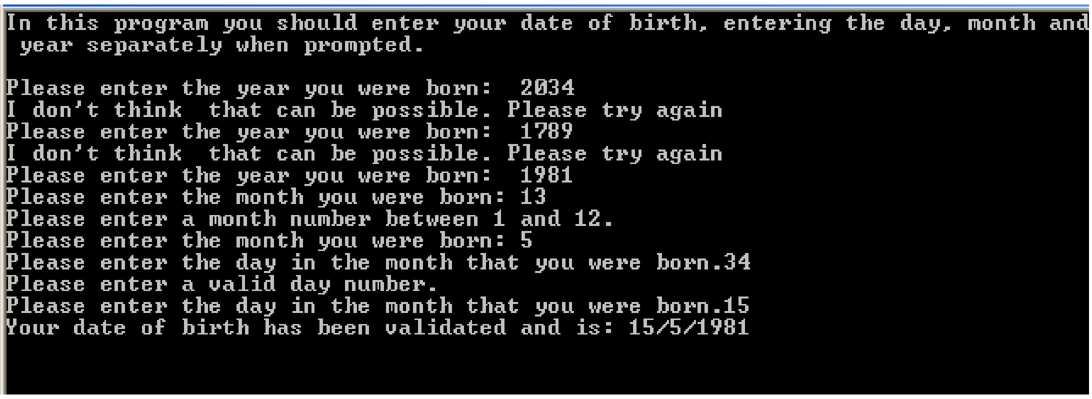
Code

Screenshots

58:DateValidation

Task

Create a function that asks the user for their date of birth as separate integers for year, then month, then day. The function validates the input to ensure that all the input is within an acceptable range. The function should loop until valid input is entered. Initially, accept days of the month between 1 and 31.



```
In this program you should enter your date of birth, entering the day, month and
 year separately when prompted.

Please enter the year you were born:  2034
I don't think  that can be possible. Please try again
Please enter the year you were born:  1789
I don't think  that can be possible. Please try again
Please enter the year you were born:  1981
Please enter the month you were born: 13
Please enter a month number between 1 and 12.
Please enter the month you were born: 5
Please enter the day in the month that you were born.34
Please enter a valid day number.
Please enter the day in the month that you were born.15
Your date of birth has been validated and is: 15/5/1981
```

Code

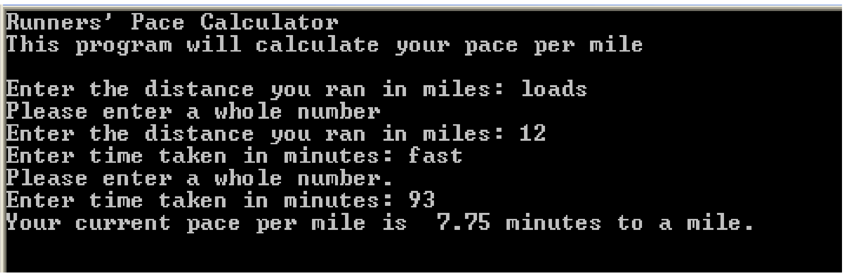
Screenshots

59: PACECALCULATOR

Task

Create a function to calculate the pace per mile for runners at a running club. The function should ask the user to input a distance, then to input the time taken, and will calculate the pace.

The function should use exception handling to trap errors in the input, for example, entering a real number or string instead of an integer. The function should loop until acceptable input is provided.



```
Runners' Pace Calculator
This program will calculate your pace per mile

Enter the distance you ran in miles: loads
Please enter a whole number
Enter the distance you ran in miles: 12
Enter time taken in minutes: fast
Please enter a whole number.
Enter time taken in minutes: 93
Your current pace per mile is  7.75 minutes to a mile.
```

Code

Screenshots

61 WORD GAMES

Task

Write a program that gets two words from the user and then displays a message saying if the first word can be created using the letters from the second word or not.

For example:

- The word EAT can be formed from the word ATE as the first word uses one E, one A and one T and the second word also contains one of each of these letters.
- The word EAT can be formed from the word HEART as the second word contains one E, one A and one T which are the letters needed to form the first word.
- The word TO can be formed from the word POSITION as the second word contains one T and (at least) one O which are the letters needed to form the first word.
- The word MEET cannot be formed from the word MEAT as the second word only contains one E and two Es are needed to form the first word.

You may assume that the user will only enter words that consist of upper case letters.

Code

Screenshots

62 Prime Numbers

Task

Write a program that checks which numbers from a series of numbers entered by the user are prime numbers.

The program should get a number from the user and then display the messages:

- "Not greater than 1" if the number entered is 1 or less
- "Is prime" if the number entered is a prime number
- "Is not prime" otherwise.

The user should then be asked if they want to enter another number and the program should repeat if they say that they do.

A prime number is a positive integer that will leave a remainder if it is divided by any positive integer other than 1 and itself.

You may assume that each number entered by the user is an integer.

If your program only works correctly for some prime numbers you will get some marks for this question. To get full marks for this question, your program must work correctly for any valid integer value that the user enters.

Code

Screenshots

63 Run Length Encryption

Task

One method that can be used to compress text data is run length encoding (RLE). When RLE is used the compressed data can be represented as a set of character/frequency pairs. When the same character appears in consecutive locations in the original text it is replaced in the compressed text by a single instance of the character followed by a number indicating the number of consecutive instances of that character. Single instances of a character are represented by the character followed by the number 1.

Figure 9 and Figure 10 show examples of how text would be compressed using this method.

Figure 9

Original text: AAARRRRGGGHH

Compressed text: A 3 R 4 G 3 H 2

Figure 10

Original text: CUTLASSES

Compressed text: C 1 U 1 T 1 L 1 A 1 S 2 E 1 S 1

What you need to do

Task 1

Write a program that will perform the compression process described above. The program should display a suitable prompt asking the user to input the text to compress and then output the compressed text.

Task 2

Test the program works by entering the text AAARRRRGGGHH.

Task 3

Test the program works by entering the text A.

Code

Screenshots