OCR GCSE Computer Science

Your notes

Additional Programming Techniques

Contents

String Manipulation

Your notes

String Manipulation

What is string manipulation?

- String manipulation is the use of **programming techniques** to **modify**, **analyse** or **extract** information **from a string**
- Examples of string manipulation include:
  - **Case conversion** (modify)
  - **Length** (analyse)
  - **Substrings** (extract)
  - **Concatenation** (modify)
  - **ASCII conversion** (analyse)

Case conversion

- The ability to **change a string from one case to another**, for example, lower case to upper case

Function	OCR exam reference	Python	Output

Uppercase	`Name = "Sarah"`
	 `print(Name.upper)`
		`Name = "Sarah"`
		 `print(Name.upper())`
			"SARAH"

Lowercase	`Name = "SARAH"`
	 `print(Name.lower)`
		`Name = "SARAH"`
		 `print(Name.lower())`
			"sarah"

Title case	`Book = "inspector calls"`
	 `print(Book.title)`
		`Book = "inspector calls"`
		 `print(Book.title())`
			"Inspector Calls"

Length

- The ability to **count the number of characters in a string**, for example, checking a password meets the minimum requirement of 8 characters

Function	OCR exam reference	Python	Output

Length	Password = "letmein"
	
	print(Password.length)
		Password = "letmein"
		
		print(len(Password))
			7

	Password = "letmein"
	
	if Password.length >= 8 then
	
	print("Password accepted")
	
	else
	
	print("Password too short")
	
	end if
		Password = "letmein"
		
		if len(Password) >= 8:
		
		print("Password accepted")
		
		else:
		
		print("Password too short")
			"Password too short"

Substring

- The ability to **extract a sequence of characters from a larger string** in order to be used by another function in the program, for example, data validation or combining it with other strings
- Extracting substrings is performed using 'slicing', using specific start and end to slice out the desired characters
- Substring is **0 indexed** (first value is 0 not 1)

Function	OCR exam reference	Python	Output
Substring	**.substring(starting character, number of characters)**	**string[start character : end character]**	

	Word = "Revision"
	
	print(Word.substring(2,3))
		Word = "Revision"
		
		print(Word[2:5])
			"vis"

	.left(number of characters)		

	Word = "Revision"
	
	print(Word.left(4))
		Word = "Revision"
		
		print(Word[:4])
			"Revi"

	.right(number of characters)		

	Word = "Revision"
	
	print(Word.right(4))
		Word = "Revision"
		
		print(Word[4:])
			"sion"

Concatenation

- The ability to **join two or more strings together** to form a single string
- Concatenation uses the '**+**' operator to join strings together

Function	OCR exam reference	Python	Output

Concatenation	FName = "Sarah"
	
	SName = "Jones"
	
	FullName = FName + SName
	
	print(FullName)
		FName = "Sarah"
		
		SName = "Jones"
		
		FullName = FName + SName
		
		print(FullName)
			"SarahJones"

	FName = "Sarah"
	
	SName = "Jones"
	
	FullName = FName + " " + SName
	
	print(FullName)
		FName = "Sarah"
		
		SName = "Jones"
		
		FullName = FName + " " + SName
		
		print(FullName)
			"Sarah Jones"

	Name = "Sarah"
	
	print("Hello, " + Name)
		Name = "Sarah"
		
		print("Hello, " + Name)
			"Hello, Sarah"

ASCII conversion

- The ability to **return an ASCII character from a numerical value** and vice versa

Function	OCR exam reference	Python	Output
ASCII conversion	print(ASC(A))	print(ord("A"))	"65"

	print(CHR(97))	print(chr(97))	"a"

Your notes

Examiner Tips and Tricks

Remember that the '+' operator is used for concatenation of strings BUT is also the mathematical operator for addition

It is important to remember, that the same operator symbol performs different roles on different types of data (integer/string)

Worked Example

A school wants to use a program to take a students first name, last name and year of entry as inputs and use them to create a username

They want the username to follow the rule:

- **Initial + First 3 letters of last name + year**

For example, a student named David Hamilton who started in 2024 would have the username:

- **DHam2024**

The algorithm has been started below:

Line	Algorithm
01	FName = input("Enter first name")
02	LName = input("Enter last name")
03	year = input("Enter year")
04	username =
05	print(username)

Use **string manipulation** to complete line **04** to create the username **[3]**

How to answer this question

- What techniques do we need to use to create the username? substring to extract the parts of the first and last name

- Concatenation to join them together

Answer

Line	Algorithm
01	FName = input("Enter first name")
02	LName = input("Enter last name")
03	year = input("Enter year")
04	username = FName.substring(0,1) + LName.substring(0,3) + year
05	print(username)

Guidance

- FName.substring(0,1) **1 mark**
- LName.substring(0,3) **1 mark**
- username = FName.substring(0,1) + LName.substring(0,3) + year **1 mark**

Your notes

File Handling

File Handling

What is file handling?

- File handling is **the use of programming techniques** to work with information stored in **text files**
- Examples of file handing techniques are:
    - **opening** text files
    - **reading** text files
    - **writing** text files
    - **closing** text files

Concept	OCR exam reference	Python
Open	file = open("fruit.txt")	file = open("fruit.txt","r")
Close	file.close()	file.close()
Read line	file.readline()	file.readline()
Write line	file.writeline("Oranges")	file.write("Oranges")
End of file	file.endOfFile()	endOfFile = False
Create a new file	newFile("Shopping.txt")	file = open("shopping.txt","w")
Append a file	n/a	file = open("shopping.txt","a")

Python example (reading data)

Employees	Text file

file = open("employees.txt", "r") # open file in read mode
endOfFile = False # set end of file to false
while not endOfFile: # while not end of file
	Greg
	Sales
	39000

```
  name = file.readline() # read line 1
  department = file.readline() # read line 2
  salary = file.readline() # read line 3
  age = file.readline() # read line 4

  print("Name: ", name) # print name
  print("Department: ", department) # print department
  print("Salaray: ", salary) # print salary
  print("age: ", age) # print age

  if name == "": # if name is empty
    endOfFile = True # set end of file to true

file.close() # close file
```
	43
	Lucy
	Human resources
	26750
	28
	Jordan
	Payroll
	45000
	31

Your notes

Python example (writing new data)

Employees	

```
file = open("employees.txt", "a") # open file in append mode
file.write("Polly\n") # write line (\n for new line)
file.write("Sales\n")
file.write("26000\n")
file.write("32\n")

file.close() # close file
```
	Greg
	Sales
	39000
	43
	Lucy
	Human resources
	26750
	28
	Jordan
	Payroll
	45000
	31
	Polly
	Sales
	26000
	32

Examiner Tips and Tricks

When opening files it is really important to make sure you use the correct letter in the open command

- "**r**" is for reading from a file only
- "**w**" is for writing to a new file, if the file does not exist it will be created. If a file with the same name exists the contents will be overwritten
- "**a**" is for writing to the end of an existing file only

Always make a backup of text files you are working with, one mistake and you can lose the contents!

Your notes

Worked Example

Use pseudocode to write an algorithm that does the following :

- Inputs the title and year of a book from the user.
- Permanently stores the book title and year to the existing text file *books.txt* **[4]**

How to answer this question

- Write two input statements (title and year of book)
- Open the file
- Write inputs to file
- Close the file

Example answer

```
title = input("Enter title")

year = input("Enter year")

file = open("books.txt")

file.writeline(title)

file.writeline(year)

file.close()
```

Guidance

- title = input("Enter title") **1 mark for both**

  year = input("Enter year")
- file = open("books.txt") **1 mark**
- file.writeline(title) **1 mark for both**

file.writeline(year)

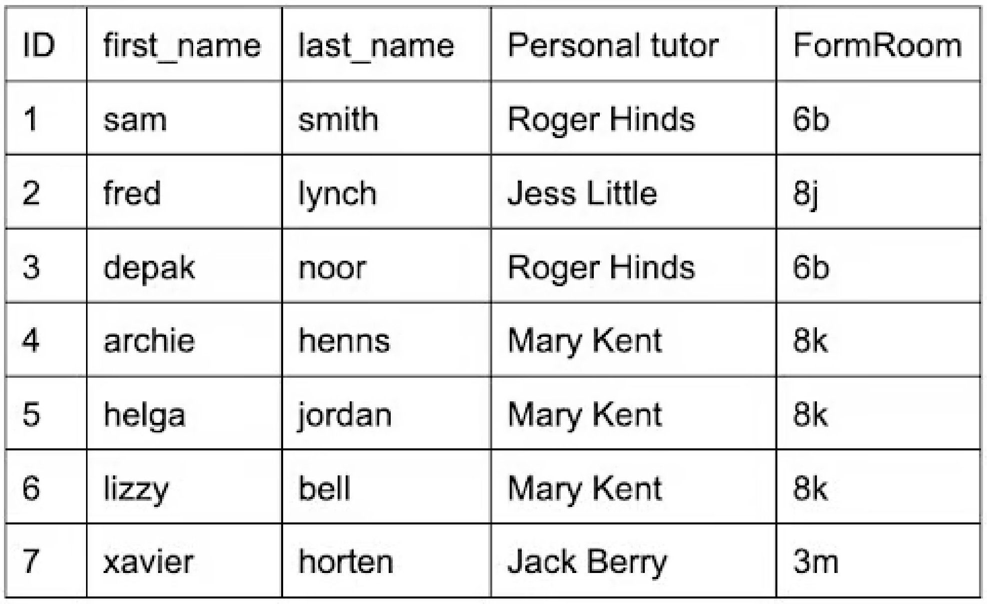
- file.close() **1 mark**

Records to Store Data

Records to Store Data

What is a database?

- A **Database** is an organised collection of data
- It allows **easy storage, retrieval, and management** of information
- A database is useful when working with l**arge amounts of data**, databases are stored on **secondary storage**
- A database is often stored on remote servers so **multiple users** can access it at the same time, useful for online systems
- Data can be **sorted and searched efficiently**, making use of more advanced structures
- They are **more secure** than text files
- A database uses **fields** and **records** to organise how it stores data



ID	first_name	last_name	Personal tutor	FormRoom
1	sam	smith	Roger Hinds	6b
2	fred	lynch	Jess Little	8j
3	depak	noor	Roger Hinds	6b
4	archie	henns	Mary Kent	8k
5	helga	jordan	Mary Kent	8k
6	lizzy	bell	Mary Kent	8k
7	xavier	horten	Jack Berry	3m

What are fields & records?

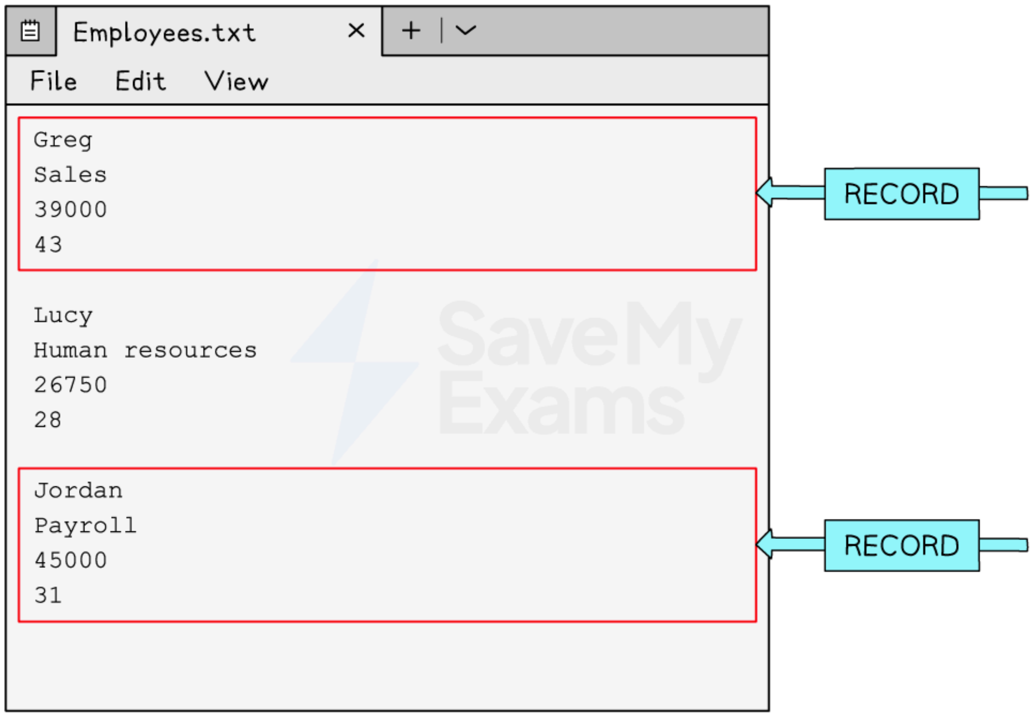
- A field is **one piece** of information relating to one person, item or object

Your notes

- A field is represented in a database by a **column**,
- A record is a **collection of fields** relating to one person, item or object
- A record is represented in a database by a **row**

Text files

- A text file is useful when working with **small amounts of data**, text files are stored on s**econdary storage** and 'read' in to a program when being used
- They are used to store information **when the application is closed**
- Each entry is stored on a new line or separated with a special identifier, for example a comma (',')
- It can be difficult in text files to know where a record begins and ends



```
Greg
Sales
39000
43

Lucy
Human resources
26750
28

Jordan
Payroll
45000
31
```

Arrays

- An array is useful when working with **small amounts of data**, arrays are stored in **main memory** (RAM)
- They are used to store information when **the application is in use**
- Can be **more efficient** and much **faster to search** than working with text files

SQL

Your notes

SQL

What is SQL?

- **SQL** (Structured Query Language) is a **programming language used to interact with** a **DBMS**.
- SQL allows users to locate specific information in a database table using these basic SQL commands:
  - **Select** - what field(s) do you want to retrieve?
  - **From** - which table(s) do you want to search?
  - **Where** - what condition is there?

Selecting Data Commands

Command	Description	Example

SELECT	Retrieves data from a database table	**SELECT** *
		(retrieves all data from the table)
		
		SELECT name, age
		(retrieves names and ages from the table)

FROM	Specifies the tables to retrieve data from	**SELECT ***
		FROM users;
		(retrieves all data from the 'users' table)
		
		SELECT name, age
		FROM users;
		(retrieves names and ages from the 'users' table)

WHERE	Filters the data based on a specified condition	**SELECT** *
		FROM users
		WHERE age > 30;
		(Retrieves all users older than 30)

- The '*****' symbol is called a '**wildcard**', it selects all fields in the table

Examples

- Select all the fields from the Customers table

Command:

```
SELECT *
FROM Customers;
```

Output:

ID	Name	Age	City	Country
1	John Doe	30	New York	USA
2	Jane Doe	25	London	UK
3	Peter Lee	40	Paris	France

- Select the ID, name & age of customers who are older than 25

Command:

```
SELECT ID, name, age
FROM Customers
WHERE Age > 25;
```

Output:

ID	Name	Age
1	John Doe	30
3	Peter Lee	40

Your notes

Worked Example

The database table Stock stores the current stock levels of products currently on sale.

ProdName	Quantity	Price
Biscuits	143	0.99
Bread	87	1.49
Milk	34	1.10
Pasta	421	0.89
Ketchup	287	2.99

Complete the SQL query to return the product name and quantity of all products that's price is less than £1 **[3]**

SELECT	
FROM	
WHERE	

Answer

SELECT	ProdName, Quantity
FROM	Stock
WHERE	Price < 1

Guidance

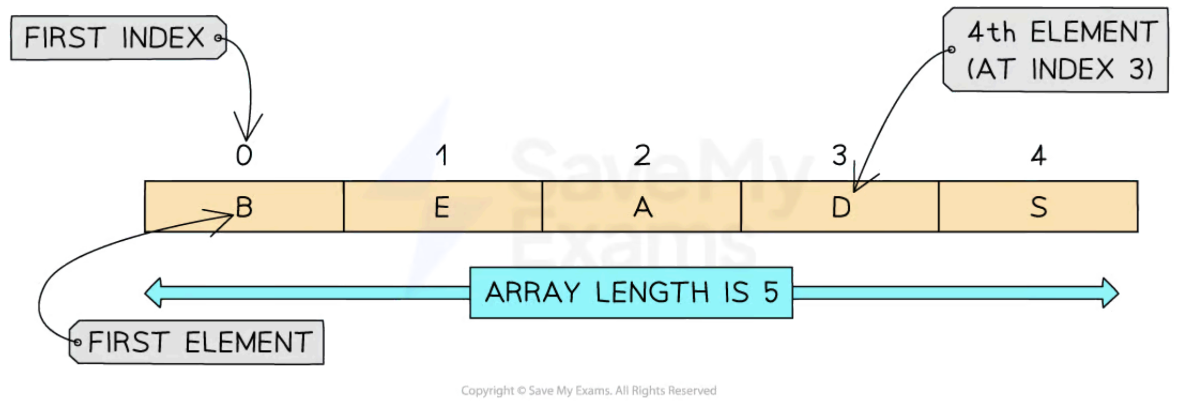
- Spelling of field names and table name must be exact
- Capitalisation must match field and table names to be awarded marks
- Table name is always in the question in different format

Arrays

1-Dimensional Arrays

What is an array?

- An array is an **ordered**, **static set of elements** in a fixed size memory location
- An array can only store **1 data type**
- A 1D array is a **linear array**
- Indexes start at 0, known as **zero indexed**



Concept	OCR exam reference	Python
Create	array scores[5]	scores = []
	Creates a blank array with 5 elements (0-4)	**Creates a blank array**
	array scores = [12, 10, 5, 2, 8]	scores = [12, 10, 5, 2, 8]
	Creates an array called scores with values assigned	
Assignment	colours[4] = "Red"	colours[4] = "Red"

	Assigns the colour "Red" to index 4 (5th element)

Your notes

Example in Python

Creating a one-dimensional array called 'array' which contains 5 integers.

- Create the array with the following syntax:
  `array = [1, 2, 3, 4, 5]`
- Access the individual elements of the array by using the following syntax:
  `array[index]`
- Modify the individual elements by assigning new values to specific indexes using the following syntax:
  `array[index] = newValue`
- Use the len function to determine the length of the array by using the following syntax:
  `len(array)`
- In the example the array has been iterated through to output each element within the array. A for loop has been used for this

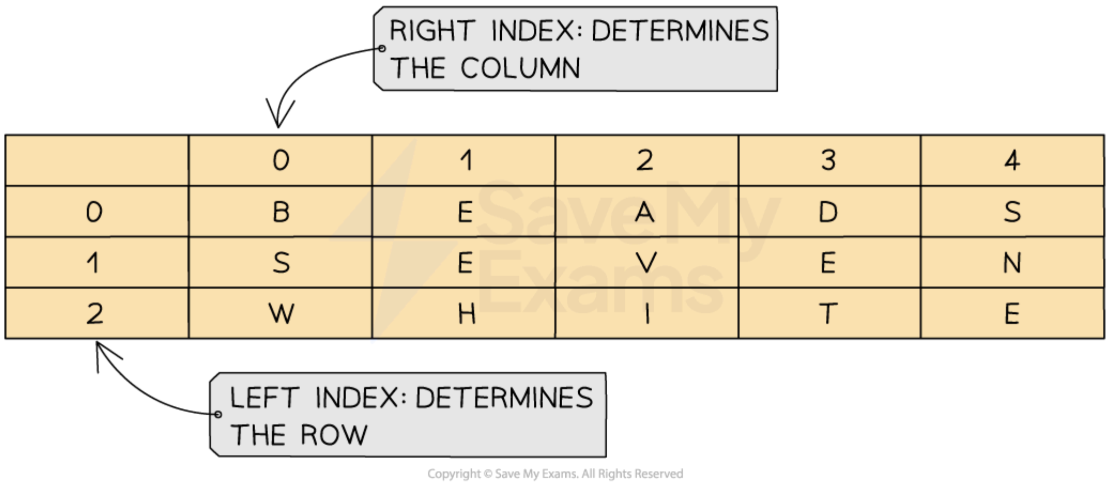
Python

```
# Creating a one-dimensional array
array = [1, 2, 3, 4, 5]

# Accessing elements of the array
print(array[0])   # Output: 1
print(array[2])   # Output: 3

# Modifying elements of the array
array[1] = 10
print(array)      # Output: [1, 10, 3, 4, 5]

# Iterating over the array
for element in array:
    print(element)

# Output:
# 1
# 10
# 3
# 4
# 5
# Length of the array
length = len(array)
print(length)     # Output: 5
```

2-Dimensional Arrays

What is a 2-dimensional array?

- A 2D array **extends** the concept on a 1D array by **adding another dimension**
- A 2D array can be visualised as **a table with rows and columns**
- When navigating through a 2D array you first have to go **down the rows** and then **across the columns** to find a position within the array


RIGHT INDEX: DETERMINES THE COLUMN
LEFT INDEX: DETERMINES THE ROW
Copyright © Save My Exams. All Rights Reserved

	0	1	2	3	4
0	B	E	A	D	S
1	S	E	V	E	N
2	W	H	I	T	E

Concept	OCR exam reference	Python
Create	array players[3,3]	players = [],[]
	Creates a blank 2D array with 3 elements (0-2)	**Creates a blank 2D array**
	players = ["Rob","Paul","Hayley"],[10, 5, 8]	players = ["Rob","Paul","Hayley"],[10, 5, 8]
	Creates a 2D array called players with values assigned	

Your notes

Your notes

Assignment	players[0,1] = "Holly"	players[0][1] = "Holly"
	Assigns the name "Holly" to index 0, 1 (1st row, 2nd column) - replaces "Paul"	

Example in Python

Python

```
# Initialising a 2D array with 3 rows and 3 columns, with the specified values
array_2d = [[1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]]

# Accessing elements in the 2D array
print(array_2d[0][0])  # Output: 1
print(array_2d[1][2])  # Output: 6
```

Examiner Tips and Tricks

In the exam, the question will always give an example to demonstrate which order the array is being read from.

Some questions can be X,Y and others can be Y, X. Always refer to the example before giving your answer!

Worked Example

A parent records the length of time being spent watching TV by 4 children

Data for one week (Monday to Friday) is stored in a 2D array with the identifier minsWatched.

The following table shows the array

	Quinn	Lyla	Harry	Elias
	0	1	2	3

Your notes

Monday	**0**	34	67	89	78
Tuesday	**1**	56	43	45	56
Wednesday	**2**	122	23	34	45
Thursday	**3**	13	109	23	90
Friday	**4**	47	100	167	23

Write a line of code to output the number of minutes that Lyla watched TV on Tuesday **[1]**

Write a line of code to output the number of minutes that Harry watched TV on Friday **[1]**

Write a line of code to output the number of minutes that Quinn watched TV on Wednesday **[1]**

Answers

- print(minsWatched[1,1] or print(minsWatched[1][1]
- print(minsWatched[2,4] or print(minsWatched[2][4]
- print(minsWatched[0,2] or print(minsWatched[0][2]

Your notes

Functions & Procedures

What are functions and procedures?

- **Functions** and **procedures** are a type of sub program, a **sequence of instructions** that **perform a specific task** or set of tasks
- Sub programs are often used to **simplify** a program by **breaking it into smaller, more manageable parts**
- Sub programs can be used to:
  - **Avoid duplicating code** and can be reused throughout a program
  - **Improve** the **readability** and **maintainability** of code
  - Perform **calculations**, to **retrieve data**, or to **make decisions** based on input
- **Parameters** are **values** that are **passed into a sub program**
  - Parameters can be variables or values and they are located **in brackets** after the name of the sub program
  - Example: `function taxCalculator(pay,taxcode)` **OR** `def taxCalculator(pay,taxcode)`
- Sub programs **can have multiple parameters**
- To use a sub program you '**call**' it from the main program

What's the difference between a function and procedure?

- A **Function returns a value** whereas a procedure does not

Concept	OCR exam reference	Python

Creating a function	`function squared(number)`
	`squared = number^2`
	`return squared`
	`endfunction`
		`def squared(number):`
		`squared = number^2`
		`return squared`

Calling a function	`SquNum = squared(4)`	`SquNum = squared(4)`

	print(SquNum)
	
	OR
	
	print(SquNum(4))
		print(SquNum)
		
		OR
		
		print(SquNum(4))

Creating a procedure	procedure ageCheck(age)
	
	if age > 18 then
	
	print("You are old enough")
	
	else
	
	print("You are too young")
	
	endif
	
	endprocedure
		def ageCheck(age):
		
		if age > 18:
		
		print("You are old enough")
		
		else:
		
		print("You are too young")

Calling a procedure	ageCheck(21)	ageCheck(21)

Examples

- A Python program using a **function to calculate area** and return the result
- Two options for main program are shown, one which **outputs the result** (# 1) and one which **stores the result** so that it can be used at a later time (# 2)

Functions

```
def area(length, width): # Function definition, length and width are parameters
  area = length * width # Calculate area
  return area # Return area

# Main program #1
length = int(input("Enter the length: ")) # Asks the user to enter the length
width = int(input("Enter the width: ")) # Asks the user to enter the width
print(area(length, width)) # Outputs the result of the function

# Main program #2
length = int(input("Enter the length: ")) # Asks the user to enter the length
width = int(input("Enter the width: ")) # Asks the user to enter the width
area = area(length, width) # Stores the result of the function in a variable
print("The area is " + str(area) + " cm^2") # Outputs the result of the function
```

Your notes

- A Python program using procedures to **display a menu** and **navigate** between them
- **Procedures** are **defined at the start of the program** and the main program calls the first procedure to start
- In this example, no parameters are needed

Procedures

```
def main_menu(): # Function definition
 print("1. Addition") # Outputs the option
 print("2. Subtraction")
 print("3. Multiplication")
 print("4. Division")
 print("5. Exit")
 choice = int(input("Enter your choice: ")) # Asks the user to enter their choice
 if choice == 1: # If the user chooses 1
  addition() # Calls the addition function
 elif choice == 2:
  subtraction()
 elif choice == 3:
  multiplication()
 elif choice == 4:
  division()
 elif choice == 5:
  exit()

def addition(): # Function definition
 num1 = int(input("Enter the first number: ")) # Asks the user to enter the first number
 num2 = int(input("Enter the second number: ")) # Asks the user to enter the second number
 print(num1 + num2) # Outputs the result of the addition

def subtraction():
 num1 = int(input("Enter the first number: "))
 num2 = int(input("Enter the second number: "))
 print(num1 - num2)

def multiplication():
 num1 = int(input("Enter the first number: "))
 num2 = int(input("Enter the second number: "))
 print(num1 * num2)

def division():
 num1 = int(input("Enter the first number: "))
 num2 = int(input("Enter the second number: "))
 print(num1 / num2)
```

```
# Main program
main_menu() # Calls the main_menu function
```

Your notes

What is a global variable?

- A global variable is **a variable declared at the outermost level of a program**. This means that they are declared outside any modules such as functions or procedures
- Global variables have a **global scope**, which means they can be **accessed and modified** from **any part of the program**

Python example

In this python code, you can see that the globalVariable (with the value 10) is declared **outside** of the function printValue. This means that this function and **any other modules** can access and change the value in the global variable

Global variables

```
globalVariable = 10 # Defines a global variable

def printValue():
    global globalVariable # Access the global variable inside a function
    print("The value into the variable is:", globalVariable)

printValue() # Call the function
```

What is a local variable?

- A local variable is **a variable declared within a specific scope**, such as a function or a code block
- Local variables are accessible **only within the block in which they are defined**, and their lifetime is limited to that particular block
- Once the execution of the block ends, **the local variable is destroyed**, and its memory is released

Python example

In this python code, you can see that the localVariable (with the value 10) is declared **inside** of the function printValue. This means that **only** this function can access and change the value in the local variable. It **cannot be accessed** by other modules in the program.

Local variables

```
def printValue():
   localVariable = 10  # Defines a local variable inside the function
   print("The value of the local variable is:", localVariable)

printValue()  # Call the function
```

Your notes

Worked Example

An economy-class airline ticket costs £199. A first-class airline ticket costs £595.

(**A**) Create a function, `flightCost()`, that takes the number of passengers and the type of ticket as parameters, calculates and returns the price to pay.

You do **not** have to validate these parameters

You must use **either**:

- OCR Exam Reference Language, **or**
- a high-level programming language that you have studied **[4]**

(**B**) Write program code, that uses `flightCost()`, to output the price of 3 passengers flying economy.

You must use **either**:

- OCR Exam Reference Language, **or**
- a high-level programming language that you have studied **[3]**

How do I answer this question?

(**A**)

- Define the function, what parameters are needed? where do they go?
- How do you calculate the price?
- Return the result

(**B**)

- How do you call a function?
- What parameters does the function need to return the result?

Answers

Part	OCR exam reference	Python

A	function flightCost(passengers, type)
	
	if type == "economy" then
	
	cost = 199 * passengers
		def flightCost(passengers, type):
		
		if type == "economy":
		
		cost = 199 * passengers

Your notes

	elseif type == "first" then
	
	cost = 595 * passengers
	
	endif
	
	return cost
	
	endfunction
		elif type == "first":
		
		cost = 595 * passengers
		
		return cost

B	print(flightCost("economy", 3)
	
	OR
	
	x = flightCost("economy", 3)
	
	print(x)
		print(flightCost("economy", 3)
		
		OR
		
		x = flightCost("economy", 3)
		
		print(x)

Your notes

Random Number Generation

Random Number Generation

What is random number generation?

- Random number generation is a programming concept that involves **a computer generating a random number** to be used within a program to add an element of **unpredictability**
- Examples of where this concept could be used include:
  - **Simulating the roll of a dice**
  - **Selecting a random question (from a numbered list)**
  - **National lottery**
  - **Cryptography**

Concept	OCR exam reference	Python

Random numbers	`number = random(1,10)`
	
	`number = random(-1.0,10.0)`
		`import random`
		
		`number = random.randint(1,10)`
		
		`number = random.randint(-1.0,10.0)`

Examples in Python

Random code

```
import random # importing random module

user = input("Enter a username: ") # asking user to enter a username
pw = input("Enter a password: ") # aksing user to enter a password

if user == "admin" and pw == "1234": # checking if the user and password are correct
  code = random.randint(1000,9999) # generating a random 4 digit code
  print("Your code is", code) # printing the code
```

National lottery

```
import random # importing random module
```

```
num1 = random.randint(1,59) # generating a random number between 1 and 59
num2 = random.randint(1,59)
num3 = random.randint(1,59)
num4 = random.randint(1,59)
num5 = random.randint(1,59)
num6 = random.randint(1,59)

print("Your lucky dip numbers are: ", num1, num2, num3, num4, num5, num6) # printing the numbers
```

Your notes