# GCSE 9-1
# COMPUTER SCIENCE

# Unit 1 - Computer systems:

## Sample exam questions

**1** The specification of two CPUs is shown in **Fig. 1**.

| Computer 1 | Computer 2 |
|---|---|
| Clock Speed: 1 GHz | Clock Speed: 1.4 GHz |
| Cache size: 2 MB | Cache size: 2 MB |
| Number of Cores: 4 | Number of Cores: 2 |

**Fig. 1**

**(a)** When running a 3D flight simulator, Computer 1 is likely to run faster than Computer 2.

Using the information in **Fig. 1**, identify **one** reason for this.

.....

..... **[1]**

**(b)** Identify **two** other parts of a computer that are not in **Fig. 1**, which could improve the performance of the computers.

1 .....

2 .....

**[2]**

**(c)** Explain **one** reason why the cache size affects the performance of the CPU.

.....

.....

.....

..... **[2]**

**(d)** **Fig. 2** lists some actions that may take place in the law company's office. Tick (✓) **one** box in each row to show which legislation applies to each action.

| Action | Data Protection Act 2018 | Computer Misuse Act 1990 | Copyright Designs and Patents Act 1988 |
|---|---|---|---|
| Using a picture for the law company's new logo without the original creator's permission. | | | |
| A secretary accessing a lawyer's personal email account without permission. | | | |
| Making a copy of the latest Hollywood blockbuster movie and sharing it with a client. | | | |
| Storing customer data insecurely. | | | |
| A lawyer installing a key logger on the secretary's computer. | | | |
| Selling client's personal legal data to a marketing company without their permission. | | | |

# Unit 1 - Computer systems:

## More sample exam questions

3 A satellite navigation system (Sat Nav) uses RAM and ROM.

**(a)** Tick (✓) **one** box in each row to show whether each of the statements is **true** for the RAM or ROM in a Sat Nav.

| | RAM | ROM |
|---|---|---|
| Stores the boot up sequence of the Sat Nav. | | |
| The contents are lost when the Sat Nav is turned off. | | |
| Holds copies of open maps and routes. | | |

The table gives the ASCII code for the characters.

| Character | ASCII code |
|---|---|
| L | 76 |
| M | 77 |
| N | 78 |
| O | 79 |
| P | 80 |

Explain how the word MOP will be represented in ASCII.

.....

.....

.....

10 A law company currently use a Local Area Network (LAN) linked to a Wide Area Network (WAN). They want to upgrade their system to utilise cloud storage.

**(a)** Define what is meant by a Wide Area Network.

..... **[1]**

**(b)** Explain **two** advantages to the law company of storing their data in the Cloud.

1 .....

.....

.....

.....

2 .....

.....

.....

.....

**[4]**

**(c)** Explain **two** disadvantages to the law company of storing their data in the Cloud.

1 .....

.....

# Unit 1 - Computer systems:

## More sample exam questions

5 **(a)** Convert the binary number 11001011 into denary.

.....

..... [1]

**(b)** Complete a 2-place shift to the right on the binary number 11001011.

.....

..... [1]

**(c)** Explain the effect of performing a 2-place shift to the right on the binary number 11001011.

.....

.....

**(b)** Identify **three** errors that the hospital staff could make that may endanger the security of the network. Outline a procedure that could be put in place to prevent each error.

Error 1 .....

Procedure 1 .....

.....

Error 2 .....

Procedure 2 .....

.....

Error 3 .....

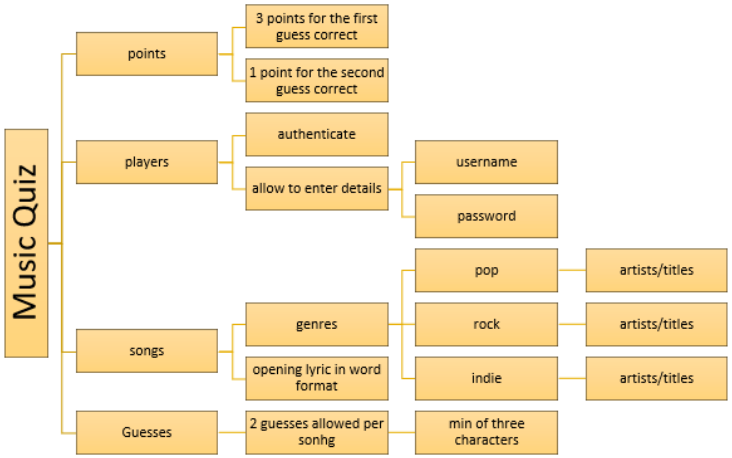
Procedure 3 .....

.....

# Unit 2 - Computational thinking, algorithms and programming:

## Sample work – breaking down a problem

Decomposition




Input and Output variables

Input Variables

- Player Identification
  - Allows the player to have an identity and be able to play the game.
  - Enter the password and username for themselves
  - The data type would be a string as the username would be a collection of characters as would the password.
  - Data validation would be a lookup check because it would have to be authenticated and presence check to check if it's there

Decomposition


# Unit 2 - Computational thinking, algorithms and programming:

## Sample work – code development

```
#re-write the new leaderboard over the old leaderboard
upload_to = open("highscores.txt","w")
upload_to.write(upload_string)
upload_to.close()
print("Leaderboard:")
print(upload_string)


while playing == True:
    while authorised == False and locked == False:
        access_code = input("Please enter your access code ")
        for x in code_list:
            if x == access_code:
                authorised == True
                game_on = True
                print("Welcome!")
            else:
                access_attempts += 1
            if access_attempts == 30: #30 because 3 attempts and 10 possible
                locked = True
                print("Too many failed attempts")
                break
```

```
Python 3.3.2 (v3.3.2:d047928ae3f6, M
(Intel)] on win32
Type "copyright", "credits" or "lice
>>> ================================
>>>
Please enter your access code ajc1
Welcome!
Please enter your access code
```

Problem: Even after entering a legitimate access code, it asks you to enter one again

Solution: Use only one equal (=) instead of a double-equals (==)

### Demonstration

```
Please enter your access code abc1
Please enter your access code ndc1
Welcome!
The song is S by Lena Meyer-Landrut. What do you think the song is? Satellite
Correct! 3 points scored
Score: 3
The next question is on its way
The song is F by Queen. What do you think the song is? Fire
Incorrect, but have another try: Flash
Correct! 1 point scored
The next question is on its way
Score: 4
The song is A by Las Ketchup. What do you think the song is? Ask
Incorrect, but have another try: Ace
Game Over. Final Score: 4
Leaderboard:
```

Incorrect access code
Correct access code

Correct first time

Correct second time

Incorrect both times

### Development

```
users= {" nat" :  1111 ,
        " jim" :  1234}
user = input("Input username")
password = input("Input password")
if user in users:
    print("Welcome back")
else:
    users[user]= password
    print ("Welcome" , user)
userpoints = 0
```

In this part of the code, it has a dictionary of pre-used names with attached passwords. It asks the user to input a name and if the name matches one in the dictionary it will welcome them back. If not, then the code will add the user to the dictionary so they are a verified player.

```
Input username nat
Input password1111
Welcome back
t
What is the song?
```

```
>>>
Input username ryan
Input password 2345
Welcome  ryan
s
What is the song?
```

```
a = open("music_songs", "a+")
a.write("thank u,next/nbury a friend/nalien boy/ngreen/noui/nshe way out/nsober/ncupid/ntongue tied/n")
a.close()

b = open("music_artists" , "a+")
b.write("ariana grande/nbillie eilish/noliver tree/ncavetown/njeremih/nthe 1975/nlorde/nryan beatty/ngrouplove")
b.close()
```

In this part of the code, it creates two files (one for songs and one for artists) and it adds all the songs. I have added limited number of songs because of the time scheme but more genres and songs could be added.

```
import random
lines = open("music_songs").read().splitlines()
myLine = random.choice(lines)
print(myLine[1])
```

This part uses the random function and opens the file for songs and splits the lines up. The random choice function is then used to pick a song at random and then it prints the first letter of the name

This part asks the person to guess the song if they get it correct

```
guess = 0
while guess <=2 :
    player_guess = input("What is the song?")
    if player_guess == myLine:
            userpoints = userpoints + 3
            print ("correct")
            print ("YOUR POINTS ARE", userpoints)
            guess = 0
            import random
            lines = open("music_songs").read().splitlines()
            myLine = random.choice(lines)
            print(myLine[1])
            if userpoints > 30:
                print("YOU WIN! YOUR POINTS ARE" , userpoints )
                break


    elif player_guess != myLine:
            guess = guess + 1
            print("try again")
            if guess == 2:
                print("game over")
                break
```

# Unit 2 - Computational thinking, algorithms and programming:

## Sample exam questions

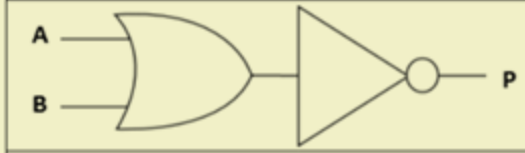
**(a)** Complete the truth table in **Fig. 1** for the Boolean statement `P = NOT (A AND B)`.

| A | B | P |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | ..... |
| 1 | 0 | ..... |
| 1 | 1 | 0 |

**Fig. 1**

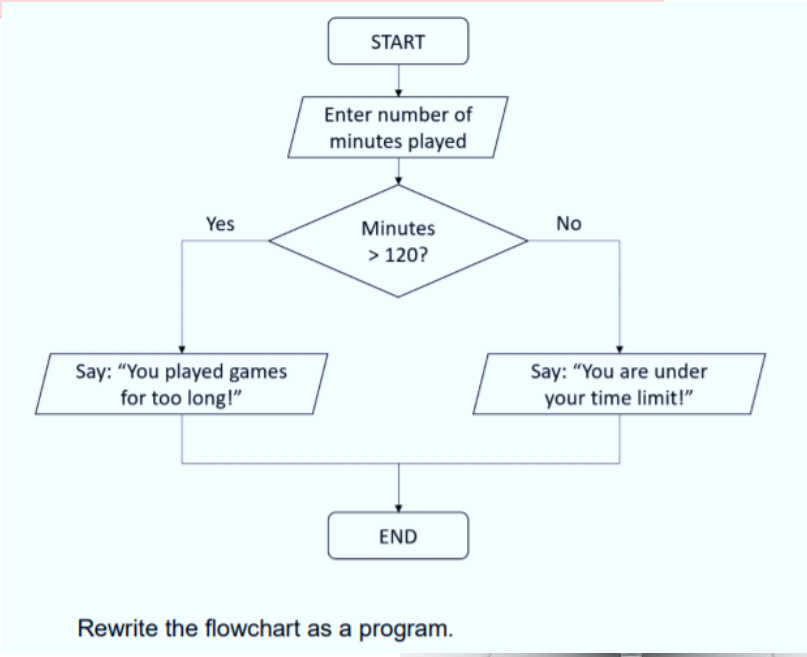
[2]

**(b)** Tick (✓) **one** box to identify the correct logic diagram for `P = NOT (A AND B)`.

| P = NOT (A AND B) | Tick (✓) one box |
|---|---|
|   | |
|   | |

**(a)** Complete the pseudocode for this program.

```
num1 = .....
num2 = input("enter second number")
..... num1 > ..... then
    .....
else
    .....
endif
```




Rewrite the flowchart as a program.

# Unit 2 - Computational thinking, algorithms and programming:

## Sample work – code testing

### Testing and Evaluation

### Test Table

| What is being tested? | Test Data | Expected outcome | Actual outcome | Action I will take |
|---|---|---|---|---|
| Allowing authorised users | Expected: "nat"<br>Null:""<br>Invalid: @1-<br>Out of bound: N/A<br>Boundary: N/A | Only accept players on the list | Allows people to join the list. | Nothing. |
| Songs chosen at random | Expected: a song<br>Null:<br>Invalid:<br>Out of bound:<br>Boundary: | It should choose a random song and not in order | It works | Nothing |
| Display of chosen song | Expected:" g___"<br>Null:""<br>Invalid:<br>Out of bound:<br>Boundary: | It should display a random song with only the first letters showing | It doesn't have the artists name and doesn't do each letter of each word | |
| Correct answer | Expected: correct<br>Null:<br>Invalid:<br>Out of bound:<br>Boundary: | It should recognise the correct answer and say correct | It says correct and moves on | |

### Evaluation

- I think overall the coding went well as I was able to use the random function however the code is unable to authenticate user as I found it to be difficult to code due to the fact that every time I coded it, it failed. Overall it went well.
- Majority of the success criteria was met and a version of the game was produced. It adds a user if the user is not on instead of authentication which means that point was partially met

### Testing

| Requirement Tested: | Pre-condition to test: | Input Type and Input Data | Expected Outcome | Actual Outcome | Changes Made |
|---|---|---|---|---|---|
| REQ_1 | n/a | n/a | "Please enter your access code" | "Please enter your access code" | None needed |
| REQ_2 | REQ_1 passed | Expected Data[1]: "abc123" | "Please enter your access code" | "Please enter your access code" | None needed |
| | | Null Data: "" | "Please enter your access code" | "Please enter your access code" | None needed |
| | | Unexpected data[1]: "ajc1" | "Welcome" | "Welcome" | None needed |
| REQ_3 | REQ_2 passed, and repeated twice | Expected Data[1]: "abc123" | "Too many failed attempts" | "Too many failed attempts" | None needed |
| | | Null Data: "" | "Too many failed attempts" | "Too many failed attempts" | None needed |
| | | Unexpected data[1]: "ajc1" | "Welcome" | "Welcome" | None needed |
| REQ_4 | REQ_1 passed | Expected Data: "ajc1" | "Welcome" | "Welcome" | None needed |
| | | Unexpected Data: "abc123" | "Please enter your access code" | "Please enter your access code" | None needed |
| REQ_5 | REQ_4 passed | Expected Data: [correct song name] | "Correct!" | "Correct!" | None needed |
| | | Unexpected Data: [incorrect song name] | "Incorrect, but have another try" | "Incorrect, but have another try" | None needed |
| | | Null Data: "" | "Incorrect, but have another try" | "Incorrect, but have another try" | None needed |
| REQ_6 | REQ_4 passed | Expected Data: [incorrect song name] | "Incorrect, but have another try" | "Incorrect, but have another try" | None needed |
| | | Unexpected Data: [correct song name] | "Correct!" | "Correct!" | None needed |
| | | Null Data: "" | "Incorrect, but have another try" | "Incorrect, but have another try" | None needed |
| REQ_7 | REQ_6 passed | Expected Data: [incorrect song name] | "Game Over" | "Game Over" | None needed |
| | | Unexpected Data: [correct song name] | "Correct!" | "Correct!" | None needed |
| | | Null Data: "" | "Game Over" | "Game Over" | None needed |