Computer Science Faculty Summer Work – 2022

We hope that you are looking forward to embarking upon your A Level Computer Science studies in September. In the meantime, we want to provide you with an opportunity to prepare for the course. You will be completing tasks which we will ask you to submit via a Google Classroom assignment at the start of the course. This will help us to assess your effort and skill, which enable us to understand your strengths and weaknesses.

The pages that follow contain three different programming challenges. You should use a programming language of your choice to create solutions for each of the three problems. We would encourage you to use repl.it (https://replit.com/) to create your programmed solutions as this will make it easy for you to share your work with us at the start of the academic year. All program code should always contain **detailed comments.**

However, we are not only interested in your programming skills. We are also interested in how you problem solve, carry out testing and can write in detail about the process you have embarked upon in creating a programmed solution. This is a significant part of the A Level Computer Science course. As such, for each of the three programming challenges you should create a report detailing the following stages:

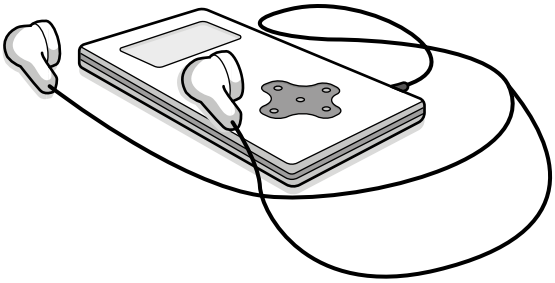
- Analysis – list the success criteria for the programme and how you will go about ensuring they are met
- Development – take screen shots of sections of your code **whilst developing your solution** and write about the programming techniques you have used and how this meets the success criteria
- Testing – create a test plan containing at least 8 tests. Carry out the tests, provide evidence (screenshots) and state whether each test passed or failed. Where tests have failed, note what corrective action has been undertaken.
- Evaluation – recap your success criteria and state how each one has been met. If they have not all been met (which is fine), state why. Be honest in your reflection - if it is your technical understanding that is fine.

Use the above bullet points as headings to help structure your written submission.

Create your write up in either a Google Doc or Word Document. One single document containing the write up for all three tasks would be preferred.

Your teacher will ask you to submit this work to a Google Classroom assignment after your first Computer Science lesson of the year. Please ensure it is ready for submission with delay.

TASK 1



Jay is creating a music quiz game.

The game stores a list of song names and their artist (e.g. the band or solo artist name). The player needs to try and guess the song name.

The game is played as follows:

- A random song name and artist are chosen.
- The artist and the first letter of each word in the song title are displayed.
- The user has two chances to guess the name of the song.
- If the user guesses the answer correctly the first time, they score 3 points. If the user guesses the answer correctly the second time they score 1 point. The game repeats.
- The game ends when a player guesses the song name incorrectly the second time.

Only authorised players are allowed to play the game.

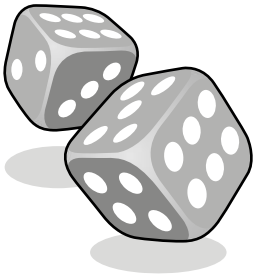
Where appropriate, input from the user should be validated.

Design, develop, test and evaluate a system that:

1. Allows a player to enter their details, which are then authenticated to ensure that they are an authorised player.
2. Stores a list of song names and artists in an external file.
3. Selects a song from the file, displaying the artist and the first letter of each word of the song title.
4. Allows the user up to two chances to guess the name of the song, stopping the game if they guess a song incorrectly on the second chance.
5. If the guess is correct, add the points to the player's score depending on the number of guesses.
6. Displays the number of points the player has when the game ends.
7. Stores the name of the player and their score in an external file.
8. Displays the score and player name of the top 5 winning scores from the external file.

Turn over

TASK 2



Ahmed is developing a two-player dice game.

The players roll two 6-sided dice each and get points depending on what they roll. There are 5 rounds in a game. In each round, each player rolls the two dice.

The rules are:

- The points rolled on each player's dice are added to their score.
- If the total is an even number, an additional 10 points are added to their score.
- If the total is an odd number, 5 points are subtracted from their score.
- If they roll a double, they get to roll one extra die and get the number of points rolled added to their score.
- The score of a player cannot go below 0 at any point.
- The person with the highest score at the end of the 5 rounds wins.
- If both players have the same score at the end of the 5 rounds, they each roll 1 die and whoever gets the highest score wins (this repeats until someone wins).

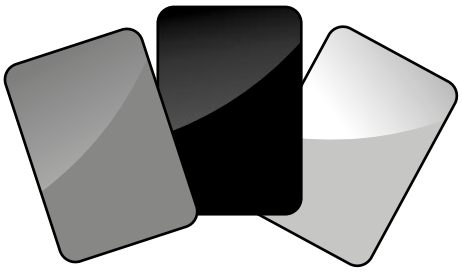
Only authorised players are allowed to play the game.

Where appropriate, input from the user should be validated.

Design, develop, test and evaluate a program that:

1. Allows two players to enter their details, which are then authenticated to ensure that they are authorised players.
2. Allows each player to roll two 6-sided dice.
3. Calculates and outputs the points for each round and each player's total score.
4. Allows the players to play 5 rounds.
5. If both players have the same score after 5 rounds, allows each player to roll 1 die each until someone wins.
6. Outputs who has won at the end of the 5 rounds.
7. Stores the winner's score, and their name, in an external file.
8. Displays the score and player name of the top 5 winning scores from the external file.

TASK 3



Joshua is creating a card game for two players.

The game uses a deck of cards. There are 30 cards in a deck. Each card has one colour (red, black or yellow). Each card has a number (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) for each colour. Each card is unique.

The 30 cards are shuffled and stored in the deck.

The rules are:

- Player 1 takes the top card from the deck.
- Player 2 takes the next card from the deck.
- If both players have a card of the same colour, the player with the highest number wins.
- If both players have cards with different colours, the winning colour is shown in the table.

Card	Card	Winner
Red	Black	Red
Yellow	Red	Yellow
Black	Yellow	Black

- The winner of each round keeps both cards.
- The players keep playing until there are no cards left in the deck.

Only authorised players are allowed to play the game.

Where appropriate, input from the user should be validated.

Design, develop, test and evaluate a program that:

1. Allows two players to enter their details, which are then authenticated, to ensure that they are authorised players.
2. Shuffles the 30 cards in the deck.
3. Allows each player to take a card from the top of the deck. Play continues until there are no cards left in the deck.
4. Calculates the winner and allocates both cards to the winner.
5. Displays which player wins (the player with the most cards).

6. Lists all of the cards held by the winning player.
7. Stores the name and quantity of cards of the winning player in an external file.
8. Displays the name and quantity of cards of the 5 players with the highest quantity of cards from the external file.