SOFE 3950: Operating Systems

Lab 3 - Jeopardy

Date: February 20th 2022

Lab Group[B]9	
Student Name	**Student ID**
Abbas Rizvi	100746798
Mohammad Hamza Asif	100767042
Matthew Gardiner	100768198

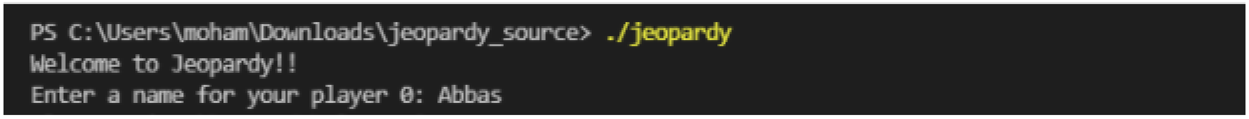
Project definition

The purpose of this tutorial activity was to collaborate on a single project to create a command line version of Jeopardy which can support up to four players with the support of github.
We compile the program using ‘gcc jeopardy.c questions.c players.c jeopardy’ and then run it with ‘./jeopardy’.

Project Requirements

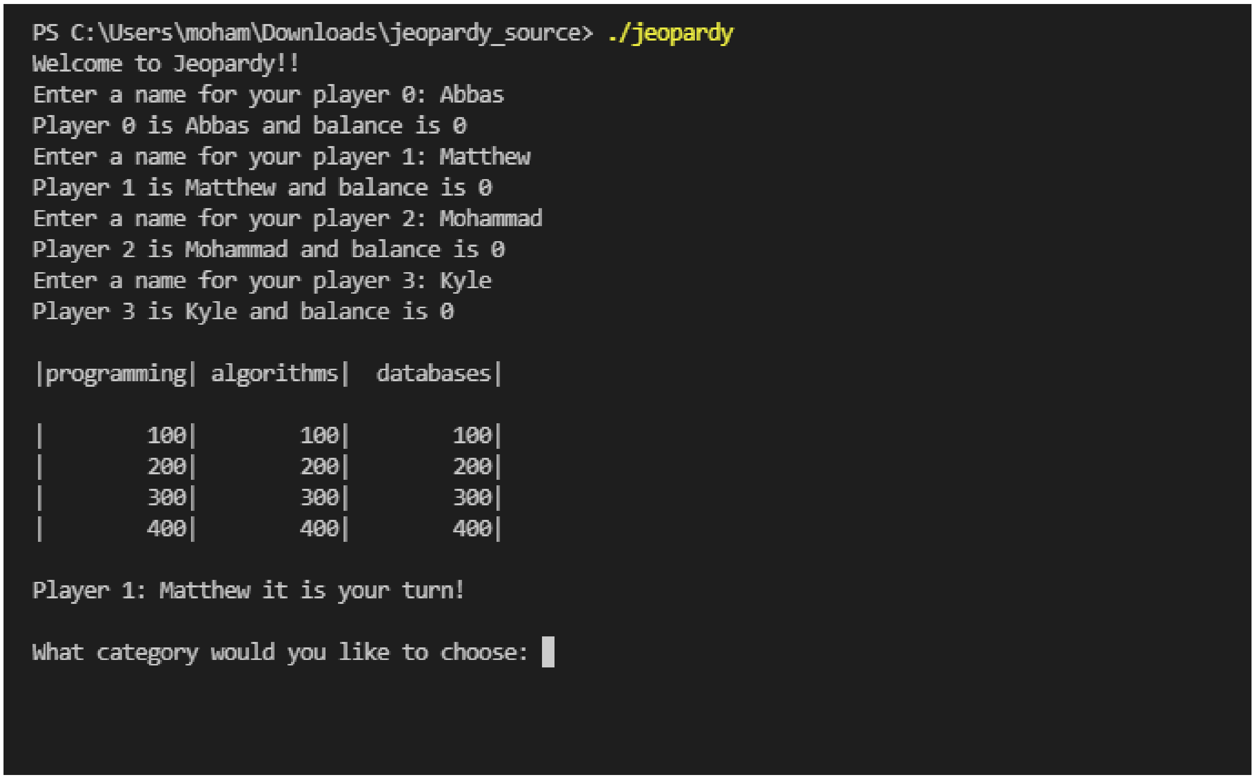
These are the main guidelines, you are free to add more than is required here (coloured output, terminal animations, etc.) to personalize your project and make it your own.

1. Your program must have a command line prompt that is used for all interactions with the Jeopardy system.



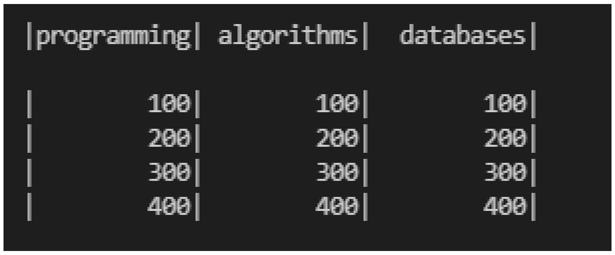
```
PS C:\Users\moham\Downloads\jeopardy_source> ./jeopardy
Welcome to Jeopardy!!
Enter a name for your player 0: Abbas
```

2. The program must ask for the names of the four players who will be playing Jeopardy, each player must enter their name and have it recorded in the system and their initial earnings set as 0.



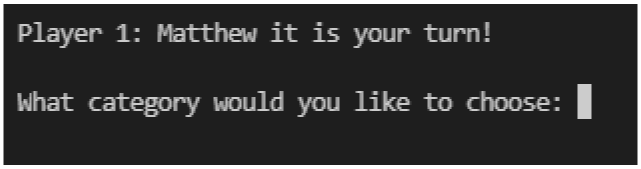
```
PS C:\Users\moham\Downloads\jeopardy_source> ./jeopardy
Welcome to Jeopardy!!
Enter a name for your player 0: Abbas
Player 0 is Abbas and balance is 0
Enter a name for your player 1: Matthew
Player 1 is Matthew and balance is 0
Enter a name for your player 2: Mohammad
Player 2 is Mohammad and balance is 0
Enter a name for your player 3: Kyle
Player 3 is Kyle and balance is 0

|programming| algorithms|  databases|

|        100|        100|        100|
|        200|        200|        200|
|        300|        300|        300|
|        400|        400|        400|

Player 1: Matthew it is your turn!

What category would you like to choose:
```

3. The program then starts the game of Jeopardy and prints out each of the question categories and displays the dollar values for each question.



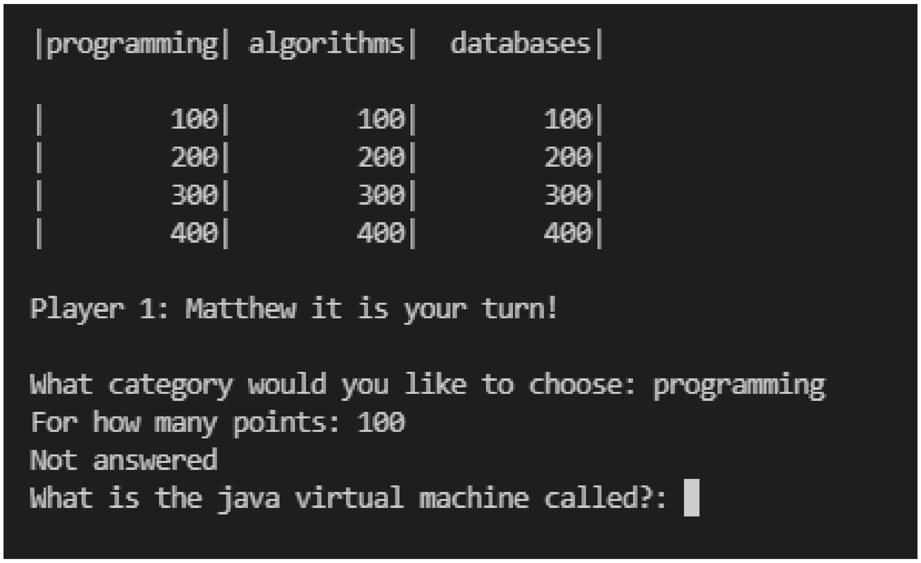
```
|programming| algorithms|  databases|

|        100|        100|        100|
|        200|        200|        200|
|        300|        300|        300|
|        400|        400|        400|
```

4. The program must accept the name of the person selected to pick the category and question.
○ You can select the person any way you choose (first person to raise their hand, random selection, etc.).
○ The program must validate that the name of the person entered is correct and matches to one of the players.
rand() was used to select a player so validation was deemed unnecessary.



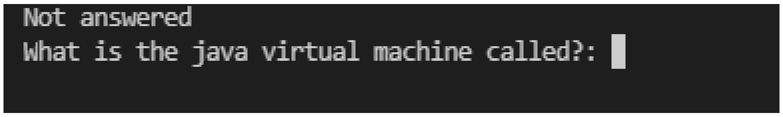
```
Player 1: Matthew it is your turn!

What category would you like to choose:
```

5. Once the player's name has been entered the program must prompt the player for the category and dollar amount question.
○ The program must verify that the category and dollar amount question has not already been answered.



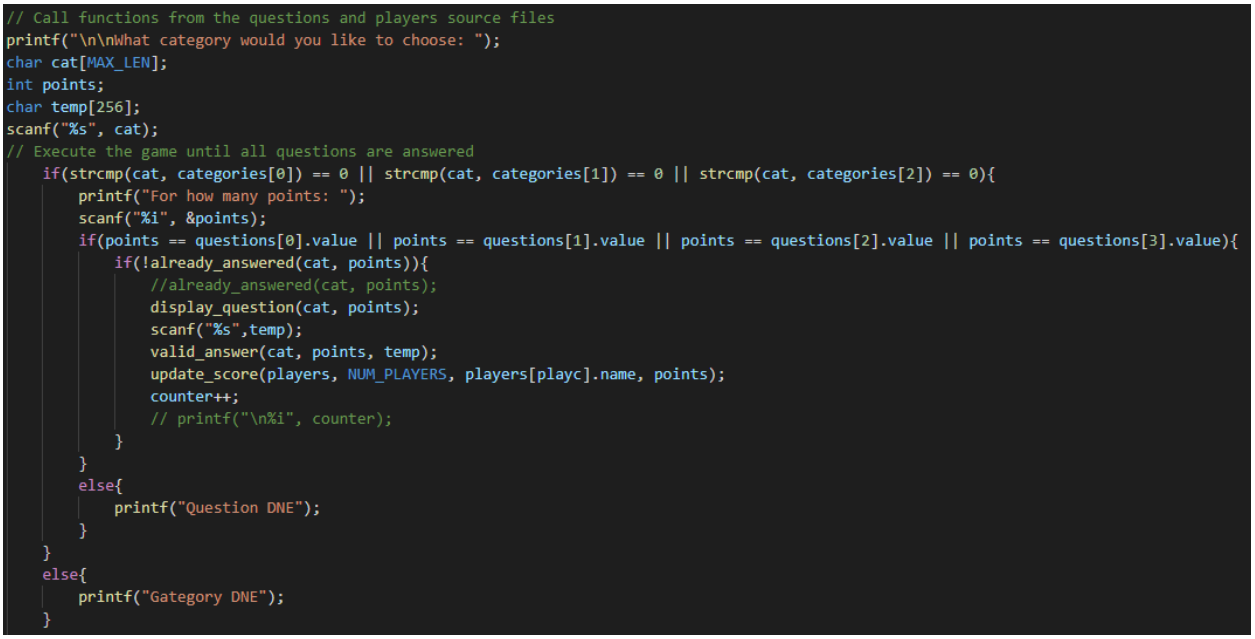
```
|programming| algorithms|  databases|

|        100|        100|        100|
|        200|        200|        200|
|        300|        300|        300|
|        400|        400|        400|

Player 1: Matthew it is your turn!

What category would you like to choose: programming
For how many points: 100
Not answered
What is the java virtual machine called?:
```

6. The program must then display the question for the player to then answer.
○ The questions and answers can be defined for each category and for each dollar amount for each category.
○ However, if you want to make the game more interesting for multiple playthroughs you can have a bank of questions and randomly select each question for each category from this question bank.



```
Not answered
What is the java virtual machine called?: 
```

7. After the question is displayed, the program must prompt the player for the answer, the player must enter the answer starting with one of the following: what is or who is.
○ You will need to use string tokenization to parse the answer, use the resources above to help.
○ To make the game easier to program you can have every question only require a one word answer.



```c
// Call functions from the questions and players source files
printf("\n\nWhat category would you like to choose: ");
char cat[MAX_LEN];
int points;
char temp[256];
scanf("%s", cat);
// Execute the game until all questions are answered
    if(strcmp(cat, categories[0]) == 0 || strcmp(cat, categories[1]) == 0 || strcmp(cat, categories[2]) == 0){
        printf("For how many points: ");
        scanf("%i", &points);
        if(points == questions[0].value || points == questions[1].value || points == questions[2].value || points == questions[3].value){
            if(!already_answered(cat, points)){
                //already_answered(cat, points);
                display_question(cat, points);
                scanf("%s",temp);
                valid_answer(cat, points, temp);
                update_score(players, NUM_PLAYERS, players[playc].name, points);
                counter++;
                // printf("\n%i", counter);
            }
        }
        else{
            printf("Question DNE");
        }
    }
    else{
        printf("Gategory DNE");
    }
```

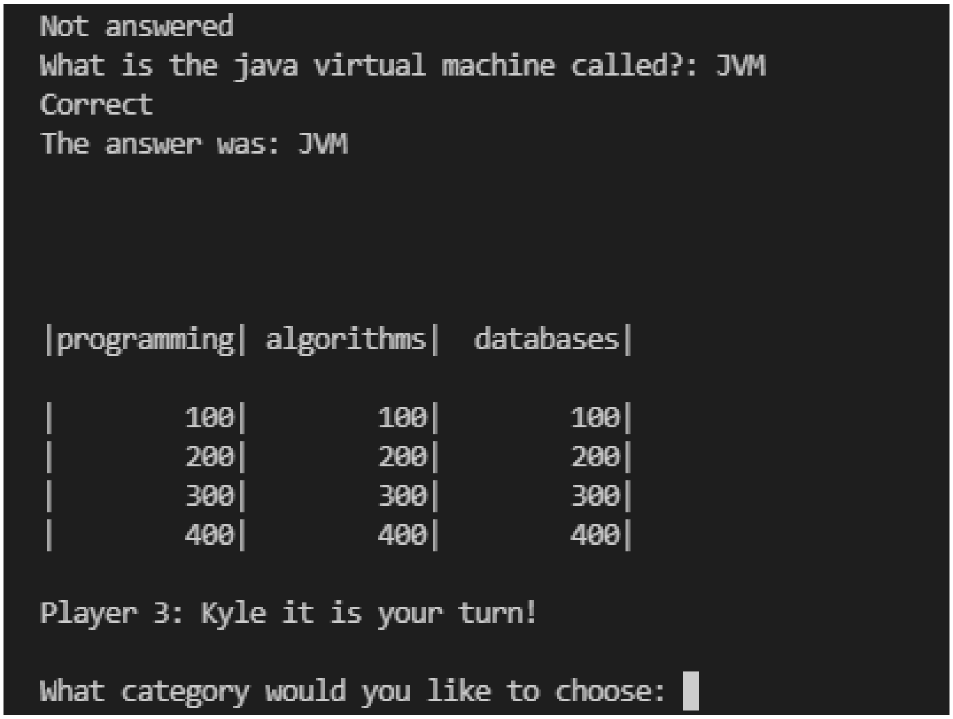
Every question in our program only requires a one word answer.

8. After the player answers the question the program must display whether they got it correct or incorrect.

○ If the player answered the question correctly the program must update their score.

○ If they answered incorrectly the program will display the answer and no user will receive any points.

○ The question must then be marked as answered so it cannot be used again.

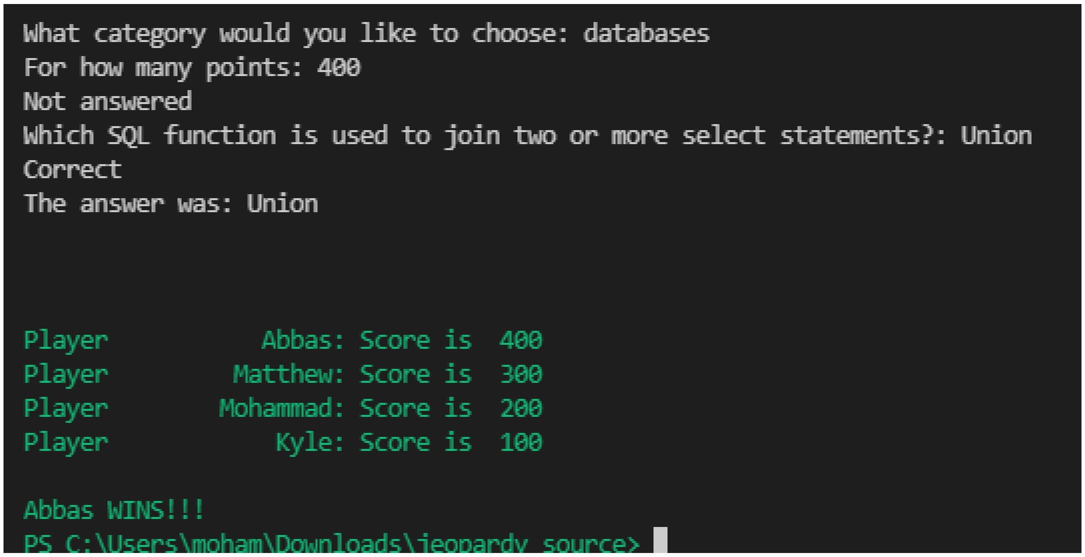


```
Not answered
What is the java virtual machine called?: JVM
Correct
The answer was: JVM




|programming| algorithms|  databases|

|        100|        100|        100|
|        200|        200|        200|
|        300|        300|        300|
|        400|        400|        400|

Player 3: Kyle it is your turn!

What category would you like to choose:
```

9. After the question has been completed, the program must print the remaining categories and question dollar amounts, then repeat requirements 4 to 9, until all of the questions have been completed.

The program repeats throughout all of the questions until 12 have been answered.

10. Once the players have completed all of the questions the game must then print each of the players from the first place to last place and print their total earnings.
○The player with the highest earnings is the winner.



```
What category would you like to choose: databases
For how many points: 400
Not answered
Which SQL function is used to join two or more select statements?: Union
Correct
The answer was: Union



Player           Abbas: Score is  400
Player         Matthew: Score is  300
Player        Mohammad: Score is  200
Player            Kyle: Score is  100

Abbas WINS!!!
PS C:\Users\moham\Downloads\jeopardy source>
```