**Project 1**



The project is to create a 2-player game of dice.

# Game Rules:

* The user selects the number of rounds to be run. Each dice round consists of 3 values:
  + **Dice** – value of the dice rolled. It is a random number between 1-6.
  + **Points** – value associated with the round. It is a random number from 10-100 in multiples of 10. (10 \* random number between 1-10).
  + **Type** – the type of the round. There are 3 types of rounds:
    - **REGULAR** – keep the “points” equal to the number of points as calculated above using a random number generator.
    - **BONUS** – replace “points” to be equal to 200.
    - **DOUBLE** – update “points” to be equal to DOUBLE the number of points as calculated above using a random number generator.
    - Probability of the round types:
      * **20% for BONUS, 30% for DOUBLE and 50% for REGULAR**
* The game consists of two players (1 & 2). The game starts with one player at random.
* For each round, one of the following cases can occur:
  + **Success:** Player-1 (odd player) is the current player and the dice rolled is odd   
    OR Player-2 (even player) is the current player and the dice rolled is even.
    - Current player gains the points, based on the type of the round (see above). The current player’s turn continues in the next round.
  + **Failure:** Player-1 (odd player) is the current player and the dice rolled is even   
    OR Player-2 (even player) is the current player and the dice rolled is odd.
    - Current player incurs penalty of the same number of points, based on the type of the round (see above). In the next round, the current player is changed to the other player.
* At the end of all rounds, the player with the highest points wins the game.

# Task 1: Complete the main.c

* Download the attached main.c
* Follow the instructions written in the comments in the main() function. The main() is the driver of the program. It calls the functions below to play the game.

# Task 2 – create and write dicegame.h

* Include the header guard in the correct format.
* Create an **enum** named **ROUNDTYPE** to identify the three round types:
  + **BONUS, DOUBLE, REGULAR**
* Write all the function prototypes – see Task below.

# Task 3 – create and write dicegame.c

Write the following functions that uses the game rules described above:

* **int getRandomNumber( int min, int max )** – This function computes a random number between min and max, **inclusive**, and returns it.
* **ROUNDTYPE getRoundType( )** – This function decides the type of the round based on the probability values mentioned in game rules (**20% for BONUS, 30% for DOUBLE and 50% for REGULAR**). HINT: Use the **getRandomNumber()** function to generate a random number between 0 through 9 and use this value for probability.
* **int getRoundPoints( ROUNDTYPE roundType )** – This function calculates and return the points associated with the given round type, based on the rules mentioned above. You will need to call the **getRandomNumber()** function to generate the points randomly.
* **void printPlayerPoints( int p1, int p2 )** – This function prints out the player points at the end of each round in the format shown below:  
  P1: 200  
  P2: 100
* **void printRoundInfo( ROUNDTYPE t, int dice, int points )** – This function prints out the information for each round in the format shown below: Type : BONUS  
  DICE : 4  
  POINTS : 200

# Submission

Be sure that your code follows the class coding style requirements. Your output should be similar in format as compared to the sample output shown below. Create a folder named your abc123, place all program files in this folder. Zip the folder and submit this abc123.zip file.

# Rubric

[10 points] – General Requirements

[3 points] – coding style - proper comments, indentation

[3 points] – correctness – compiles properly and gives correct output

[4 points] – submission – no missing files, zip, properly submitted

[16 points] - main.c file

[2 points] - initialize the random number generator

[2 points] - define and initialize all required variables

[2 points] - get the number of rounds from the user

[8 points] - game loop with function calls and proper point calculations

[2 points] - determine and print the final outcome when the game is over

[8 points] - dicegame.h

[2 points] - header guard

[2 points] - enum

[4 points] - function prototypes

[16 points] – dicegame.c()

[3 points] - getRandomNumber() – calculates and returns a random number

[4 points] - getRoundType() – returns the round type based on probability

[5 points] - getRoundPoints() – returns points for a round given the round type

[2 points] - printPlayerPoints() – prints points for both players in the shown format

[2 points] - printRoundInfo() – prints info for given round in the shown format

# Sample Output

## ############### RUN-1 ###############

Enter the number of rounds: 5

P1 : 0

P2 : 0

ROUND 1

--------

Player : 1

Type : REGULAR

Dice : 5

Points : 70

P1 : 70

P2 : 0

ROUND 2

--------

Player : 1

Type : REGULAR

Dice : 2

Points : 50

P1 : 20

P2 : 0

ROUND 3

--------

Player : 2

Type : DOUBLE

Dice : 4

Points : 180

P1 : 20

P2 : 180

ROUND 4

--------

Player : 2

Type : DOUBLE

Dice : 3

Points : 140

P1 : 20

P2 : 40

ROUND 5

--------

Player : 1

Type : DOUBLE

Dice : 3

Points : 180

P1 : 200

P2 : 40

GAME OVER!!

P1 Won

## ############### RUN-2 ###############

Enter the number of rounds: 5

P1 : 0

P2 : 0

ROUND 1

--------

Player : 2

Type : REGULAR

Dice : 6

Points : 40

P1 : 0

P2 : 40

ROUND 2

--------

Player : 2

Type : BONUS

Dice : 3

Points : 200

P1 : 0

P2 : -160

ROUND 3

--------

Player : 1

Type : BONUS

Dice : 4

Points : 200

P1 : -200

P2 : -160

ROUND 4

--------

Player : 2

Type : REGULAR

Dice : 5

Points : 20

P1 : -200

P2 : -180

ROUND 5

--------

Player : 1

Type : DOUBLE

Dice : 6

Points : 100

P1 : -300

P2 : -180

GAME OVER!!

P2 Won