In [ ]:
```
/*Reches P. Eric K.
27/09/2023
Lab 3 1.0*/
```

In [1]:
```
/*
We start by including the libraries needed to run the program.
You won't need to know all the contents of these, and some libraries
are only needed to run some pre-written code you don't have to handle.
*/
#include <iostream>
#include <vector>
#include <algorithm>
#include <random>
#include <ctime>
using namespace std; //this line makes some of the code ccshorter

//This code is here to set up the dice box and randomize its contents
vector<char> dice_box = {'R','R','R','G','G','G','G','G','G','Y','Y','Y','Y'};
srand(time(NULL));
auto rng = default_random_engine {};
shuffle(begin(dice_box), end(dice_box), rng);

//this code is here to track any of the dice rolled to escape
vector<char> escape_dice;
```

In [2]:
```
/*/*
Given a character representing the color of a die, this function prints
out the word for that color. There are only 3 possibilities:
printDieColor('R') prints out "red"
printDieColor('G') prints out "green"
printDieColor('Y') prints out "yellow"
*/
/*void printDieCoulour(char colour){
    if (toupper(colour) == 'R'){
        printf("red");
    }
    else if (toupper(colour) == 'G'){
        printf("green");
    }
    else if (toupper(colour) == 'Y'){
        printf("yellow");
    }
}*/
```

In [3]:
```
/*
Given a character representing the result of a die, this function prints
out the word for that result. There are only 3 possibilities:
printDieResult('B') prints out "brain"
printDieResult('E') prints out "escape"
printDieResult('H') prints out "hit"
*/
/*void printDieResult(char result){
    if (toupper(result) == 'B'){
        printf("brain");
    }
    else if (toupper(result) == 'E'){
        printf("escape");
    }
    else if (toupper(result) == 'H'){
        printf("hit");
    }
}*/
```

In [4]:

```
/*
Given which die was rolled (1,2, or 3), the result, and the color, print out
a message indicating all of the information about the roll.
For example, printDieInfo(2, 'B', 'G') should print the following:

rolling die #2...
You rolled a brain on a green die

*/
/*void printDieInfo(int number, char result, char colour){
    printf("rolling die #%d...\n",number);
    printf("You rolled a ");
    printDieResult(result);
    printf(" on a ");
    printDieCoulour(colour);
    printf(" die\n");


}*/
```

In [5]:

```
//printDieInfo(1,'b','g')
```

In [6]:

```
/*The rollDie function rolls a die and returns the result,
represented by a character. It has a single parameter,
which is a character representing the color of the die.
The color can be red ('R'), green ('G'), or yellow ('Y').
The 6 green dice have 3 brains, 1 hit and 2 escapes,
the 4 yellow dice have 2 of each,
and the 3 red dice have 1 brain, 3 hits and 2 escapes.

The possible return values for rollDie include:
'H' = Hit
'E' = Escape
'B' = Brain
*/
//TODO: Complete the rollDie function!
/*char rollDie(char colour){
    char result;
    int num = rand() % 6;
    if (toupper(colour) == 'G'){
        if(num == 0 or num == 1 or num == 2){
            result = 'B';
        }
        else if(num == 3 or num == 4){
            result = 'E';
        }
        else if(num == 5){
            result = 'H';
        }
    }
    else if (toupper(colour) == 'Y'){
        if(num == 0 or num == 1){
            result = 'B';
        }
        else if(num == 2 or num == 3){
            result = 'H';
        }
        else if(num == 4 or num == 5){
            result = 'E';
        }
    }
    else if (toupper(colour) == 'R'){
        if(num == 0){
            result = 'B';
        }
        else if(num == 1 or num == 2 or num == 3){
            result = 'H';
        }
        else if(num == 4 or num == 5){
            result = 'E';
        }
    }
    return result;
}*/
```

In [7]:

```
//rollDie('g')
```

In [8]:

```
/*
Call this function any time you need to refill the dice in the dice box
and shuffle its contents.
*/
void resetDiceBox(){
    escape_dice.clear();
    dice_box = {'R','R','R','G','G','G','G',
                              'G','G','Y','Y','Y','Y'};
    auto rng = std::default_random_engine {};
    std::shuffle(std::begin(dice_box), std::end(dice_box), rng);
}
```

In [9]:

```
//This function returns true if the box is empty, and false otherwise.
bool diceBoxIsEmpty(){
    return dice_box.size() == 0;
}
```

In [10]:

```
/*
This function takes a die from the dice box
and returns a character representing the color of the die taken
from the dice box.
'R' = Red
'G' = Green
'Y' = Yellow
*/
char takeDie(){
    if (diceBoxIsEmpty()){
        return '0';
    }
    int elem = rand()%dice_box.size();
    char die = dice_box[elem];
    vector<char>::iterator iter1 = dice_box.begin();
    for (int i = 0; i < elem; i++){
        iter1++;
    }
    dice_box.erase(iter1);
    return die;
}
```

In [11]:

```
/*
The rollDie function rolls a die and returns the result,
represented by a character. It has a single parameter,
which is a character representing the color of the die.
The color can be red ('R'), green ('G'), or yellow ('Y').
The 6 green dice have 3 brains, 1 hit and 2 escapes,
the 4 yellow dice have 2 of each,
and the 3 red dice have 1 brain, 3 hits and 2 escapes.

The possible return values for rollDie include:
'H' = Hit
'E' = Escape
'B' = Brain
*/
char rollDie(char colour){
    char result;
    int num = rand() % 6;
    if (toupper(colour) == 'G'){
        if(num == 0 or num == 1 or num == 2){
            result = 'B';
        }
        else if(num == 3 or num == 4){
            result = 'E';
        }
        else if(num == 5){
            result = 'H';
        }
    }
    else if (toupper(colour) == 'Y'){
        if(num == 0 or num == 1){
            result = 'B';
        }
        else if(num == 2 or num == 3){
            result = 'H';
        }
        else if(num == 4 or num == 5){
            result = 'E';
        }
    }
    else if (toupper(colour) == 'R'){
        if(num == 0){
            result = 'B';
        }
        else if(num == 1 or num == 2 or num == 3){
            result = 'H';
        }
        else if(num == 4 or num == 5){
            result = 'E';
        }
    }
    return result;
}
```

In [12]:
```
/*
Given a character representing the color of a die, this function prints
out the word for that color. There are only 3 possibilities:
printDieColor('R') prints out "red"
printDieColor('G') prints out "green"
printDieColor('Y') prints out "yellow"
*/
void printDieCoulour(char colour){
    if (toupper(colour) == 'R'){
        printf("red");
    }
    else if (toupper(colour) == 'G'){
        printf("green");
    }
    else if (toupper(colour) == 'Y'){
        printf("yellow");
    }
}
```

In [13]:
```
/*
Given a character representing the result of a die, this function prints
out the word for that result. There are only 3 possibilities:
printDieResult('B') prints out "brain"
printDieResult('E') prints out "escape"
printDieResult('H') prints out "hit"
*/
void printDieResult(char result){
    if (toupper(result) == 'B'){
        printf("brain");
    }
    else if (toupper(result) == 'E'){
        printf("escape");
    }
    else if (toupper(result) == 'H'){
        printf("hit");
    }
}
```

In [14]:
```
/*
Write a function called finishRound which takes any parameters needed, and
resets all the necessary variables as well as indicate that the round is over.

It should make use of the resetDiceBox() function.

You will need to make finishRound a pass-by-reference function!
*/
void finishRound(int &current_player, int &hits, int &brains, int &previous, int &cont
    printf("round over\n");
    current_player = 3 - current_player; //1 for player 1, 2 for player 2
    hits = 0; //the number of hits so far rolled by the current player
    brains = 0; //the number of brains so far rolled by the current player
    previous = 0;
    cont = 0;
    resetDiceBox();

}
//TODO: write the finishRound function!
```

In [15]:
```
/*
This function will print that the game has finished, and indicate
the winner, or if the game is a draw.
*/
void endGame(int player1_brains, int player2_brains){
    printf("Game Over!\n");
    if (player1_brains > player2_brains){
        printf ("Player 1 wins!\n");
    } else if (player1_brains < player2_brains){
        printf ("Player 2 wins!\n");
    } else {
        printf("It's a draw!\n");
    }
}
```

In [16]:
```
/*
Given which die was rolled (1,2, or 3), the result, and the color, print out
a message indicating all of the information about the roll.
For example, printDieInfo(2, 'B', 'G') should print the following:

rolling die #2...
You rolled a brain on a green die

*/
void printDieInfo(int number, char result, char colour){
    printf("rolling die #%d...\n",number);
    printf("You rolled a ");
    printDieResult(result);
    printf(" on a ");
    printDieCoulour(colour);
    printf(" die\n");



}
```

In [17]:
```
/*
This function sets aside a die that was just rolled with an 'escape' result.
It is necessary to use this in order for takeEscapeDie() to function correctly.
*/
void setAsideEscapeDie(char color){
    escape_dice.insert(escape_dice.begin(), color);
}
```

In [18]:
```
/*
This function retrieves one of the dice rolled with an "escape" result
in the previous throw.
```

```
*/
char takeEscapeDie(){
    if (escape_dice.size() == 0){
        return '0';
    }
    char result = escape_dice.back();
    escape_dice.pop_back();
    return result;
}
```

In [19]:

```
/*
This function will play the Zombie Dice game.
It will rely on all the previous functions to work correctly.

TODO: Add the missing code to this function!

*/
void playGame(){
    int current_player = 1; //1 for player 1, 2 for player 2
    int player1_brains = 0; //the total sum of brains scored so far by player 1
    int player2_brains = 0; //the total sum of brains scored so far by player 2
    int hits = 0; //the number of hits so far rolled by the current player
    int brains = 0; //the number of brains so far rolled by the current player
    bool is_final_turn = false; //true only when we reach the final turn
    int previous = 0;
    int cont = 0;
    resetDiceBox();
    char color;


    printf("Time for a new game!\n");
    while (true){
        printf("It's Player %d's turn!\n", current_player);

       cont = 0;
        for (int i = 1; i < 4; i++){
            //TODO: Missing line of code: get a new die and check its color!
            if (previous > 0){
                color = takeEscapeDie();
                previous -= 1;
            }
            else{
                color = takeDie();
            }
            if (color != '0'){
                //TODO: Missing line of code: roll the die and get the result!
                char result = rollDie(color);

                printDieInfo(i,result,color);

                if (result == 'B'){
                    brains++;
                } else if (result == 'H'){
                    hits++;
                }else if (result == 'E'){
                    setAsideEscapeDie(color);
                    cont++;
                }
            } else {
                printf("There are no more dice left in the box!\n");
            }
        }
        previous += cont;
        if (hits >= 3){
            printf("Knocked out! No brains this round :(\n");
            //TODO: Missing line of code: call the finishRound function
            finishRound(current_player, hits, brains, previous, cont);

            if (is_final_turn){
                endGame(player1_brains, player2_brains);
                break;
            }
        }

        else {
            printf("You've acquired %d brain(s) this round.\n ", brains);
            printf("You've acquired %d hit(s). \n", hits);
            printf("Do you want to keep rolling? Y/N");
            char answer;
            cin >> answer;
            if (answer == 'N' or answer == 'n'){
                //add brains to the current player's score
                if (current_player == 1){
                    player1_brains += brains;
                } else {
                    player2_brains += brains;
                }

                //TODO: Missing line of code: call the finishRound function
                finishRound(current_player, hits, brains, previous, cont);

                printf("Player 1 has %d brains and Player 2 has %d brains.\n",
                       player1_brains, player2_brains);

                if (is_final_turn){
                    endGame(player1_brains, player2_brains);
                    break;
                }
                else if (player1_brains > 12 || player2_brains > 12){
                    is_final_turn = true;
                    printf("Player %d has reached 13 brains!\n", 3 - current_player);
                    printf("Player %d gets one more turn!\n", current_player);
                }
            }
        }
    }
}
```

In [20]:

```
playGame();
```

```
Time for a new game!
It's Player 1's turn!
rolling die #1...
You rolled a escape on a yellow die
rolling die #2...
You rolled a brain on a yellow die
rolling die #3...
You rolled a brain on a green die
You've acquired 2 brain(s) this round.
 You've acquired 0 hit(s).
Do you want to keep rolling? Y/N

It's Player 1's turn!
rolling die #1...
You rolled a brain on a yellow die
rolling die #2...
You rolled a brain on a yellow die
rolling die #3...
You rolled a brain on a green die
You've acquired 5 brain(s) this round.
 You've acquired 0 hit(s).
Do you want to keep rolling? Y/N

It's Player 1's turn!
rolling die #1...
You rolled a escape on a yellow die
rolling die #2...
You rolled a escape on a red die
rolling die #3...
You rolled a escape on a green die
You've acquired 5 brain(s) this round.
 You've acquired 0 hit(s).
Do you want to keep rolling? Y/N

It's Player 1's turn!
rolling die #1...
You rolled a brain on a yellow die
rolling die #2...
You rolled a escape on a red die
rolling die #3...
You rolled a brain on a green die
You've acquired 7 brain(s) this round.
 You've acquired 0 hit(s).
Do you want to keep rolling? Y/N

It's Player 1's turn!
rolling die #1...
You rolled a escape on a red die
rolling die #2...
You rolled a brain on a green die
rolling die #3...
You rolled a escape on a green die
You've acquired 8 brain(s) this round.
 You've acquired 0 hit(s).
Do you want to keep rolling? Y/N
```

```
It's Player 1's turn!
rolling die #1...
You rolled a brain on a red die
rolling die #2...
You rolled a brain on a green die
rolling die #3...
You rolled a hit on a red die
You've acquired 10 brain(s) this round.
 You've acquired 1 hit(s).
Do you want to keep rolling? Y/N

round over
Player 1 has 10 brains and Player 2 has 0 brains.
It's Player 2's turn!
rolling die #1...
You rolled a escape on a green die
rolling die #2...
You rolled a brain on a green die
rolling die #3...
You rolled a brain on a green die
You've acquired 2 brain(s) this round.
 You've acquired 0 hit(s).
Do you want to keep rolling? Y/N

round over
Player 1 has 10 brains and Player 2 has 2 brains.
It's Player 1's turn!
rolling die #1...
You rolled a brain on a yellow die
rolling die #2...
You rolled a escape on a red die
rolling die #3...
You rolled a escape on a red die
You've acquired 1 brain(s) this round.
 You've acquired 0 hit(s).
Do you want to keep rolling? Y/N

It's Player 1's turn!
rolling die #1...
You rolled a hit on a red die
rolling die #2...
You rolled a escape on a red die
rolling die #3...
You rolled a hit on a green die
You've acquired 1 brain(s) this round.
 You've acquired 2 hit(s).
Do you want to keep rolling? Y/N
```

```
round over
Player 1 has 11 brains and Player 2 has 2 brains.
It's Player 2's turn!
rolling die #1...
You rolled a brain on a yellow die
rolling die #2...
You rolled a hit on a red die
rolling die #3...
You rolled a brain on a yellow die
You've acquired 2 brain(s) this round.
 You've acquired 1 hit(s).
Do you want to keep rolling? Y/N

round over
Player 1 has 11 brains and Player 2 has 4 brains.
It's Player 1's turn!
rolling die #1...
You rolled a escape on a red die
rolling die #2...
You rolled a escape on a green die
rolling die #3...
You rolled a escape on a yellow die
You've acquired 0 brain(s) this round.
 You've acquired 0 hit(s).
Do you want to keep rolling? Y/N

It's Player 1's turn!
rolling die #1...
You rolled a escape on a red die
rolling die #2...
You rolled a hit on a green die
rolling die #3...
You rolled a brain on a yellow die
You've acquired 1 brain(s) this round.
 You've acquired 1 hit(s).
Do you want to keep rolling? Y/N

It's Player 1's turn!
rolling die #1...
You rolled a escape on a red die
rolling die #2...
You rolled a brain on a green die
rolling die #3...
You rolled a brain on a yellow die
You've acquired 3 brain(s) this round.
 You've acquired 1 hit(s).
Do you want to keep rolling? Y/N
```

```
round over
Player 1 has 14 brains and Player 2 has 4 brains.
Player 1 has reached 13 brains!
Player 2 gets one more turn!
It's Player 2's turn!
rolling die #1...
You rolled a brain on a red die
rolling die #2...
You rolled a hit on a yellow die
rolling die #3...
You rolled a escape on a green die
You've acquired 1 brain(s) this round.
 You've acquired 1 hit(s).
Do you want to keep rolling? Y/N

round over
Player 1 has 14 brains and Player 2 has 5 brains.
Game Over!
Player 1 wins!
```

In [ ]: