Jingsheng Zhao

Hailey Wilder

Eric Durboraw

Michael Ross

Andrew Schapper

Group Project

**DESIGN:**

This program will play a game of Rock, Paper, Scissors. The program will contain 5 class files, three of which would be called Rock, Paper, and Scissors. The fourth class is the game class whcih contains the actual game play features. Lastly, the fourth class will be the tool class. This class will have an int function called strength and a char function called type. It also contains a void function called setStrength() which takes in the int to set the strength of the tool. The other three class will inherit from the Tool class. All of these class will contain a default constructor that will set the strength to 1 and a non-default constructor that will take in an int that is used to initialize the strength field. This constructor will also initialize the type field using ‘r’ for Rock, ‘p’ for Paper, and ‘s’ for Scissors.

The ways that this program will compare the strengths of each tool will be by having a bool fight(Tool). This will use these limitations

* Rock's strength is doubled (temporarily) when fighting scissors, but halved (temporarily) when fighting paper
* In the same way, paper has the advantage against rock, and scissors against paper
* The strength field shouldn't change in the function, which returns true if the original class wins in strength and false otherwise.

We will be also implementing a game class which will allow the user to play the rock, paper, or the scissor against the computer. It will also contain two Tool \* and three int fields to help keep track of the number of human wins, computer wins and ties.

Each of the other classes will be set up with the following functions.

**Game Class:**

Protected:

* chooseStrength();
* MatchCheck() ;
* humanWins;
* compWins;
* ties;
* rockStrength;
* paperStrength;
* ScissorsStrength;

Public:

* ~Game();
* Game(const Game &other);
* void operator=(const Game &other);
* int GamePlay();

**Rock/Paper/Scissors Class:**

These classes will be set up similarly. They will contain the same types of functions. I will demonstrate what the functions will look like with the rock class.

Public:

* Rock();
* ~Rock();
* Rock(const Rock &);
* bool fight(Tool);
* Rock(int);
* void operator=(const Rock &other);

Lastly, the layout for the tool class will look like this and its important to show this because each class as stated above is going to inherit from this class

**tool class:**

protected:

* int strength;
* char type;

public:

* Tool();
* void setStrength(int);
* virtual bool fight(Tool);
* virtual ~Tool();
* void setType(int);
* int getType();
* int getStrength();

We will also be implementing a main file to combine all of these classes and files together in order to run the program.

**Testing**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class | Test Case | Input Values | Driver Functions | Expected Outcome | Observed outcome |
| Game | GameMenu1 | Y | GamePlay() | Set strength | Infinite error loop |
| Game | GameMenu2 | N | GamePlay() | Continue to the game | Infinite error loop |
| Game | GameMenu3 | 5 | GamePlay() | Reprompt for a number in range | Exception thrown |
| Game | GameMenu3 | -1 | GamePlay() | Reprompt for a number in range | Exception thrown |
| Game | CompPick1 | N/A | CompPick() | Generate a random number between 1 and 3 and assign a tool | Consistent output P, P, R, P,S,P. Rand() not seeded. |
| Game | CompPick2 | N/A | CompPick() | Generate a random number between 1 and 3 and assign a tool | Random output after seeding |
| Game | chooseStrength1 | -1 | chooseStrength() | Error | Accepted. Opposite of expected result (i.e. rock beats paper) |
| Game | chooseStrength2 | -1 for rock, paper, and scissors strength | chooseStrength() | error | As expected |
| Game | Toolinput1 | r | Toolinput() | Chooses rock | As expected |
| Game | Toolinput2 | S | Toolinput() | Chooses scissors | As expected |
| Game | Toolinput3 | P | Toolinput() | Chooses paper | As expected |
| Game | Toolinput4 | E | Toolinput() | exits | Segmentation fault |
| Game | MatchCheck1 | Win | MatchCheck() | Increments humanWins and displays victory message | As expected |
| Game | MatchCheck2 | Loss | MatchCheck() | Increments compWins and displays victory message | As expected |
| Game | MatchCheck3 | Tie | MatchCheck() | Increments ties and displays a tie message | As expected |