cGroup Project Reflection

Group Name: Group 16

Members:

Mahtab Brar, Craig Ricker, Hisami Scott, & Melissa Nardone

Design Description:

Programs:

Tool.cpp	*Tool.h*	*Rock.cpp*	*Rock.h*
Paper.cpp	*Paper.h*	*Scissors.cpp*	*Scissors.h*
RPSGame.cpp	*RPSGame.h*	*play_game.cpp*	*Menu.cpp*
Menu.h			

Tool Class:

Tool class is a purely abstract class. This class will be created in order for other tools to e created (rock, scissors etc)

Method/Data field	Explanation
Protected	
Int _strength	The “strength” of the tool, larger number = stronger
Char _type	Type of tool, to be overwritten by child tool classes
Public:	
Virtual _setStrength(int)	Setting function for _strength
Virtual static fight()	
Constructor(int)	Sets strength to int

Paper Class: inherits from Tool class

No additional functionality needs to be added, just create a constructor which sets the strength, and the type.

Method/Data field	Explanation
Public	

Constructor()	Sets _stregth to default, and _type to “p”
Constructor(int)	Sets _stregth to input, and _type to “s”

Scissor Class: inherits from Tool class
No additional functionality needs to be added, just create a constructor which sets the strength, and the type.

Method/Data field	Explanation
Public	
Constructor()	Sets _stregth to default, and _type to “s”
Constructor(int)	Sets _stregth to input, and _type to “s”

Rock Class: inherits from Tool class
No additional functionality needs to be added, just create a constructor which sets the strength, and the type.

Method/Data field	Explanation
Public	
Constructor()	Sets _stregth to default, and _type to “r”
Constructor(int)	Sets _stregth to input, and _type to “s”

RPS_Game Class
RPS contains the logic of how different tools interact with one another. There is a human player, and an AI player.

Method/Data field	Explanation
Private	
Tool * _playerH	Human player, pointer to the tool object. User will be prompted to select tool type
Tool * _playerAI	Ai player, will be selected with some AI (randomly??)
Int _playerHScore	Human player score

Int _playerAIScore	AI score
Int _ties	Total number of ties
Void printScores()	Prints out the number of ties, and each player's scores
Public	
Void play(char)	Takes input of type char, which dictates which move the player is going to make. Points _playerH tool to a new tool of this type Points *playerAI to a tool of random type Have the winner decided, whether it is a function call to “fights” or an overloading of the operator. Update scores accordingly Make a call to printScores() for scores to be displayed

Pseudo code:

```
Main:
Print menu
Options: 0 chose rock, 1 scissor, 2 paper, 3 quit
Call RPS_Game.play(choice) if not quiting
Repeatedly print/prompt menu until quit is selected

RPS_Game
Print score:
“Human player has “ playerH points “ and Ai player has “ aiScore “. There are also “ ties “ total ties!”

play(char)
_AITool = randomTool
_Playertool = Tool_Type_Charinput
Players.fight
    -   Takes two Tool *
    -   Based on rock paper scissor etc update strength as described by the homework
        assignment
    -   Once strengths are updated, see which has highest strength
    -   Return pointer to winner
_printScore() - output scores
```

Test Plan:

1. We will test each function independently, ensuring that they all work
2. Below is a list of both the test case description, and the result

Test Results:

Test Case ID	Test case description	Test case data set Expected result	Result
1	Menu works properly Valid scenario	Input 0 - Play Input 1 - Quit	Pass
2	Menu works properly Invalid scenario Entering invalid selection	Input 3 Message displays to ask user re-enter the selection	Pass
3	Menu works properly Invalid scenario Entering non-integer type	Input "paper" Message displays to ask user re-enter the selection	Pass
4	Menu works properly Invalid scenario Entering non-integer type	Input 1.2 Message displays to ask user re-enter the selection	Pass
5	Correct tool is selected for User	Input 0 Paper is selected Input 1 Scissor is selected Input 2 Rock Rock is selected	Pass
6	Fight result is correct and properly displayed	User:paper Computer : paper, Result: tie User: paper Computer: scissor Result: Computer win User: paper Computer: rock Result: Human win (the same way for user: scissor, user: rock)	Pass

7	The current status for the wins and ties are correct and and properly displayed	You selected: paper Computer selected: scissor Computer won: Human wins: 0 Computer wins:1 Ties: 0	Pass
8	The message if user wants to play again is displayed and the selection works properly	Do you want to play again? Input 0 - Play Input 1 - Quit	Pass

Design Changes:

We changed Paper, Scissor, and Rock constructors to entirely rely on the Tool constructor. Based on each class, we pass the corresponding char and strength to the Tool constructor

We initially had troubles with memory leaks, we were not calling the delete for RPSTool object. We also needed to ensure that the Tool * within the RPSTool destructors are deleted properly.

We made play a static Tool function, instead of having a function for rock, scissor and paper. The function implementation was already a semi layer violation, but now it works on any tools and we do not need to repeatedly overwrite virtual function.