Objected-oriented Programming Report

Assignment #1

|  |
| --- |
| **Professor**  김영민 교수님 |
| **Department**  Computer engineering |
| **Student ID** 2013722004 |
| **Name** 최민기 |
| **Class** 설계 : 1(Mon) , 2(Wed) / 실습 : 3, 4 (Thu) |
| **Date**  2017. 3. 31 (Fri) |

1. Write a program that draws a rhombus as described below. The size N (odd number), given by a user, is a row of the rhombus. Display the rhombus with the number and fill out the empty space with ‘ ‘. The number increases from zero as move to the center of the rhombus and then decreases to zero as shown below.

**Result Screen)**

|  |  |
| --- | --- |
| N: 5 | N: 7 |
|  |  |
| 5 rows of rhombus | 7 rows of rhombus |

**Explanation)**

This Program receives the input odd number from the user and the input doo number becomes a line and print rhombus. The configuration of rhombus is as follows. On the center axis, the number on the left increases and the number on the right decreases. It also prints blanks to make a rhombus shape on the top and bottom based on the middle line.

**Consideration)**

First, to divide the rhombus use two while statements. In the first while statement, I print the top including the middle line. In the second while statement, I printed the bottom without the middle line. I used three for statements in the while statement. The left part printed to increase the number and right part printed so that the number decreased. I also printed blanks to make the rhombus shape based on the middle line. At first I did not get a sense. However, writing code was not difficult after I thought I would catch the middle axis.

1. Write a program that generates random numbers. If you input N(integer), program generates N random numbers (1~20) Then, count and sort out descending order how many times the number has been printed. If there are the same numbers of times, the bigger random number comes first.

**Result Screen)**

|  |
| --- |
| Input N: 10 |
|  |
| I got 10 random numbers and sort them in descending order |
| Input N: 20 |
|  |
| I got 20 random numbers and sort them in descending order |

**Explanation)**

This program generates a random number(1 to 20) as much as the user’s input value(N). And prints the frequency of the generated random number in descending order. If the frequency is the same, Prints from a large number.

**Consideration)**

I put random numbers(1 to 20) into an array that was dynamically allocated to fit the user input. And frequencies(1 to 20) stored in other array. And using the while statement, the frequency was arranged in descending order. It was easy to descend. However, when the frequency is the same, It was difficult to print from a large number. This section was resolved by printing the stored array of frequencies from 1 to 20 backwards. It was difficult and long to think about this point.

1. Write a program that performs the BINGO game. Rules of the BINGO game are as follows. The program stores numbers from 1 to 25 on 5x5 bingo board. There are two bingo boards: user bingo board and computer bingo board. When the BINGO game is started, the user and the computer alternately call numbers one by one. The called number is checked in each bingo board and displayed to ‘#’. Whenever the user or the computer calls the number, the program checks whether each bingo board is BINGO. If one horizontal line or one vertical line is all checked, it is BINGO. If one bingo board is BINGO, the game ends.

**Result Screen)**

|  |
| --- |
| <Initial Screen> |
|  |
| Initial Screen. |
|  |
| User enters 4. You can see that 4 has been changed ‘#’ on the bingo board. |
|  |
| Computer enter 12. You can see that 12 has been changed to ‘#’ on the bingo board. |
|  |
| User enter 21. You can see that 21 has been changed to ‘#’ on the bingo board. |
| Computer enter 2. You can see that 2 has been changed to ‘#’ on the bingo board. |
|  |
| User enter 16. You can see that 16 has been changed to ‘#’ on the bingo board. |
| Computer enter 5. You can see that 5 has been changed to ‘#’ on the bingo board. |
|  |
| After repeating this process, the user is able to get the result that the user has won because user’s horizontal line is bingo. |

**Explanation)**

This program is an designed to enable computer and user to play bingo. First 5 x 5 user and computer bingo boards are filled with random numbers ( 1 to 25 ). And user and computer alternately call a number. At this time, duplicate numbers con not be recalled. If user and computer first obtain vertical or horizontal bingo, It’s a winner. If they are bingo at the same time, both win.

**Consideration)**

I did not know shuffle at first making bingo board and mixing value. I was able to study concept of shuffle while doing <1-3> questions. The rest went well in order. However, It was difficult to handle the fact that computer should not be called a duplicated number when computer called a value. I solved this fact in this way that 1 to 25 in order that they are stored in the array, if the user or computer called the value of the array was treated as a value of 0. This part was the hardest and took a long time.

1. The function checks whether the first argument (str1) is equal to the second argument (str2) or not. Then, write a program that reads two strings which the lengths are less than or equal to 64 (including null character) from a console window and checks whether the two input strings are equal or not by using the function. The program should read two strings only and should print one of "They are equal." and "They are different." according to the result of the function. Note that 1) do not copy any character of string into other variables in **CompareString()**; 2) do not use any standard library function in **CompareString()**; 3) **CompareString()**is case-insensitive.

**Result Screen)**

|  |
| --- |
|  |
| I got the result that two strings are the same regardless of case |
|  |
| I got the result that two strings are the different. |

**Explanation)**

This program is receives two strings from the user. When comparing two strings, compare case-insensitive. Also do not use any standard library function in CompareString().

**Consideration)**

This question <1-4> is not difficult. When two strings are input and two strings are compared, compare with ASCII code values.

1. Write a program that reads an array of digits from the input file (‘input.txt’) and finds the maximum possible sum of two numbers made of input digits. Two numbers must have the same number of digits. All input digits should be used to make the two numbers. The first line of input consists of an integer N, where N is the size of array and even number (N < 20). The second line of input contains N array elements (from 1 to 9) separated by space. Print the result of calculation to the console window.

**Result Screen)**

|  |  |
| --- | --- |
| input.txt | console |
| N: 4  input:3 1 1 9 |  |
| Result that extract 2 from 4’s input values and add the largest value. | |
| input.txt | console |
| N: 6  input:5 8 3 2 5 1 |  |
| Result that extract 2 from 6’s input values and add the largest value. | |
| input.txt | console |
| N: 3  Input: 2 9 7 |  |
| Result that error, because of N is odd number. | |

**Explanation)**

This program reads the file(input.txt) and create the largest value by adding two numbers consisting of input values.

**Consideration)** To get the largest result of the sum of the two values when we combine the input values, I sorted first in descending order. When making the descending results into two numbers, I multiply the square of 10 to set the number of digits. It was difficult and time consuming to write this part.

1. Write a program that calculates the number of 2xn rectangles filled with 2x1or 1x2rectangles. For example, if we say n = 4, there are five number of cases as shown in the following figure. **Note: The output value is the remainder of the number of cases divided by 1000**

**Result Screen)**

|  |
| --- |
| Test Case : 7 |
|  |
| When the test case is 7, you can see that result of inputting n value 7 times and outputting appropriate output for each n value. |

**Explanation)**

This program calculate the number of 2xn rectangles filled with 2x1 rectangles. Also, the test case is input as much as the input value.

**Consideration)**

Depending on each n value, I calculate the number of 2xn rectangles filled with 2x1 rectangles. When I calculated it, I found that there was a rule and it was a Fibonacci sequence. When Fibonacci sequences were implemented as recursive functions, the n values became larger and the execution speed was very slow. So I learned about the speed difference for Fibonacci sequences. Therefore, it is implemented through loop statement so that output value can be outputted immediately even if n value increases.

**Reference**)

Assignment\_lecture\_1\_ver1 ( 김영민 교수님, 한만기 조교님, 고은정 조교님 )

Assignment\_lecture\_2\_ver1 ( 김영민 교수님, 한만기 조교님, 고은정 조교님 )

Assignment\_lecture\_3\_ver1 ( 김영민 교수님, 한만기 조교님, 고은정 조교님 )