Tatyana Vlaskin

Week 5

**Problem 1**:

In this problem will need to write a simple function that acts like a random number generator and looks like:

void rand\_int(const int &min, const int &max, int &val)

This function will accept references to min and max and return val, which is a random number generated by the random number generator

We are allowed to reuse code from the previous assignments. This was straightforward problem because I already had code from last 2 assignments; there is not much that I can say for design of this problem. See implementations for more information.

The only thing that I say is that I used const in the function definition because I did not want the function to change the min and max values. If they are not defined as const, they can be potentially changed in the function calling. I still do not understand how that would be possible, but it does happen, so I defined them as const.

Also, just for completeness, I’ll mention that when you pass a variable by value, the value gets copied. When you pass a variable by reference, the variable is stored in the address that was set aside for the variable during declaration/definition. Copying values takes more space; thus, this plays a significant role when values are huge.

**Problem 2:**

For this problem we need to write a simple function that accepts 2 references to strings and returns whether they have the same contents. As for the previous problem we are allowed to reuse code from last week, so its just a matter of defining a function that accepts 2 strings as references and slightly rearranging the code. See implementation for more information. I’ve just taken the following code from my last weeks problem and moved it to a function: void userInput(string& string1, string& string2);

string string1, string2;// strings declaration

cout << "Input string # 1: "; // console output

getline(cin, string1); // console input string 1

cout << endl;

cout << "Input string # 2: "; //console output

getline(cin, string2); //console input string 2

cout<<endl;

stringCompare(string1, string2);

**Problem 3:**

Understanding:

We need to write a program where the user inputs 3 integerst

Once integers are entered they are sorted from smaller to large value

Not required, but I am adding this to the program.

Check for valid entries. User will not be allowed to enter special characters and letters. Hower, user will be allowed to enter floating-point number, but they will be converted to interest.

Implementation:

On page 150 in our book, there is a swap function:

24 void swapValues(int& variable1, int& variable2)

25 {

26 int temp;

27 temp = variable1;

28 variable1 = variable2;

29 variable2 = temp;

30 }

I’ll try to use it for this problem. I not sure that it will work, but it looks really promising.

Lets say that the user needs to enter only 2 numbers. Ill declare those values as x (first value) and y (second value).

The user enters 2 UNSORTED values and we need to sort them from smaller to larger.

Scenario 1: x < y nothing needs to be done because value 1 is already smaller than value 2, so they are sorted.

Scenario 2: x > y. In this case we need to swap x and y.

I’ll introduce variable double temp and the following loop to swap x and y:

if (x > y){

temp= x;

x= y;

y= temp;

}

Thus, if value1 (x) > value2(y), after this loop, values will be swapped and sorted from smallest to largest. Ill put this code in a function: void Sort1(double &x, double &y);

I’ve tested it and it worked well- the code does what is expected.

The problem is that we have 3 variables, so we need to compare value 1, value 2 and value 3. We have 3 combinations that we need to compare and sort

1. Value1 and value2
2. Value 1 and value3
3. Value 2 and value 3

I’ll just write one more sort function and call sort1 3 times. Something like this:

void sort (int &value1, int &value2, int &value3){

Sort1(value1, value2);

Sort1(value1, value3);

Sort1(value2, value3);

}

**Problem 4:**

Understanding:

I read the problem like 10 times. Everything was making sense and at the same time nothing was making any sense. I gave up and decided to read more about the Finonacci number. According to the http://www.mathsisfun.com/numbers/fibonacci-sequence.html

The Fibonacci Sequence is the series of numbers:

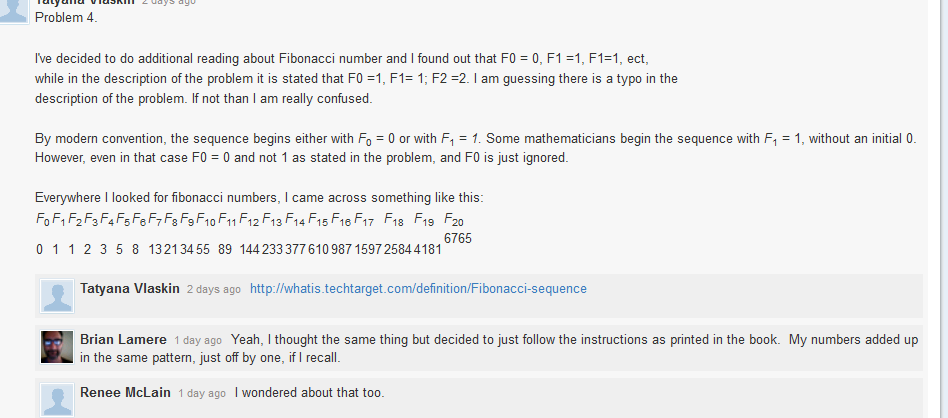
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

The next number is found by adding up the two numbers before it.

The Rule is **xn = xn-1 + xn-2**. I will definitely use this in my code. But instead of x, Ill say Fibonacci or something like that.

I found out that F0 = 0, F1 =1, F1=1, ect,

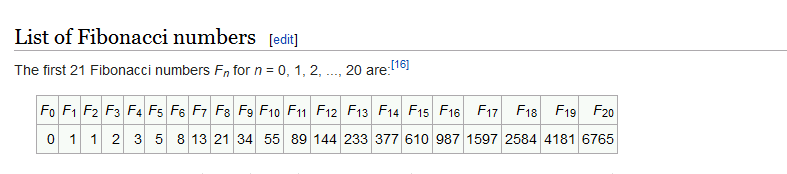
while in the description of the problem it is stated that F0 =1, F1= 1; F2 =2. I am guessing there is a typo in the description of the problem. If not than I am really confused. I’ve posted this question on the discussion board, lon Thursday, I have not gotten an answer from TA or instructor. I am attaching a snapshot:



I’ve also found information that by modern convention, the sequence begins either with *F*0 = 0 or with *F*1 = *1.* Some mathematicians begin the sequence with *F*1 = 1, without an initial 0. However, even in that case F0 = 0 and not 1 as stated in the problem, and F0 is just ignored.

Because I think that this assignment has a type, I decided to use this table to test my program.

It is taken from: http://en.wikipedia.org/wiki/Fibonacci\_number



PLEASE NOTE IN MY SOLUTION TO THIS PROBLEM F0 = 0 and not 1.

From the definition of the Fibonacci number, we have out reqursive property:

fibonacci(n-1) + fibonacci(n-2);

**PROBLEM 5:**

We need to write a program that inputs number of rows and lets us know how many pins are there in the pyramid.

Trying to find a pattern:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of rows** | **Total # of Pins in Triangle** | **Trying to find a pattern** | **Formula** | **Spaces before pins**  **(I need this to display the triangle on the screen)** |
| 1 | 1 | 1 | Base case | 0 |
| 2 | 3 | 1+2 | CountPins(2)  =  2 + CountPins(1) | 1 |
| 3 | 6 | 1+2+3 | CountPins(3)   = 3 + CountPins(2) | 2 |
| 4 | 10 | 1+2+3+4 | CountPins(4)   = 4 + CountPins(3) | 3 |
| 5 | 15 | 1+2+3+4+5 | CountPins(5)   = 5 + CountPins(4) | 4 |
| 6 | 21 | 1+2+3+4+5+6 | CountPins(6)   = 6 + CountPins(5) | 5 |
| **countPins(rows)   = rows + CountPins(rows-1)** | | | | **spaces = rows - 1** |

So basically to calculate total number of pins in the triangle, I simply add the row number to the already existing pins. This is how we figured out the general formula. Now let us look at the simplification of one of the cases: 4 rows.

Next part of the code in my program is not required, but when I saw solution for this program that Frank L Brasington posted on discussion board, I’ll decided to understand his code and incorporate it in my code to completeness of this program in case I will use this report to study for the test.

**Problem 6:**

We need to design a program which will let the user to enter principle balance, interest rate and number of years that they are planning to keep the money in the band and output the amount of money that will be in the bank at the end.

Trying to find a pattern:

|  |  |  |  |
| --- | --- | --- | --- |
| **Year** | **Amount of money in the bank**  **Lets say interest rate is 10%** | **Trying to find pattern** | **Formula** |
| 0 | 1000 $ | 1000 | Base case |
| 1 | 1100 $ | 1000 + 0.1 \*1000 | Amount(1)  =  Amount(0) + Amount (0) \* (IR/100) |
| 2 | 1210 $ | 1100 + 0.1\*1100 | Amount(2)  =  Amount(1) + Amount (1) \* (IR/100) |
| 3 | 1331 | 1210 + 0.1 \*1210 | Amount(3)  =  Amount(2) + Amount (2) \* (IR/100) |
| 4 | 1464.1 | 1331 +0.1 \*1331 | Amount(4)  =  Amount(3) + Amount (3) \* (IR/100) |
| Amount( years)  =  Amount(years-1) + Amount (year-1) \* (IR/100) | | | |

Thus our recursive function will use to following property to calculate the balance in the bank at the end of each year:

Amount( years)  =  Amount(years-1) + Amount (year-1) \* (IR/100)

Recursive function look somewhat like this:

double amount( double &principle, double rate, double years )

{

principle = principle + (principle\* (rate/ 100));

if(years > 0.0) // if there are still years left

{

principle = amount( principle, rate, (years-1.0)); // call of the function

}

return principle; // once years <=0, total amount is calculated.

}

**Problem 7**

**PROJECT**:

**UNDERSTANDING:**

1. User 1 enters a word
2. If word entered by user 1 is valid, word is stored in a variable and user 2 is asked to guess a word
3. If word entered by user 1 is not valid , user 1 get a message that entry is invalid and they are prompted to reenter a word
4. Once the word is valid and stored as a variable:
5. User 2 is notified how many guesses they have
6. Word that is entered by user one is presented to the user 2 as ‘-‘ . Each ‘-‘ represents a letter in the word
7. User 2 is asked to guess a word one letter at a time
8. If the user guesses a letter, program shows the user 2 what of the word has been guessed so far. At this step, I will also change the list of possible letters to reflect the user’s newest correct guess. If the letter was guessed correctly, ‘-‘ character will be replaced with the correct letter.
9. If the user makes incorrect guess, the user is notified that they made incorrect guess and number of misses is counter. I think I’ll have two variable misses and guesses and during implementation decide what exactly I will do with them
10. If the user makes a guess that is correct but is the repeated guess of the same character, the user is notified that the letter was already guessed. I think in this case I will not consider this as an incorrect guess, but I might change my plan.
11. We need to check if the guess is a valid letter, I have no idea what is means. I interpretation will be that is the user inters a letter that is in the word, it is a valid letter. However if the user enters a letter that is not in the word or number, or special character, the user 2 will be notified that the guess made is wrong.
12. If the user guesses a word congratulation message is displayed.
13. If the user runs out of guesses, the user is notified that they lost, what the secret word was and what letters were missing from their guesses.
14. At the end of the game the user is asked if they want to repeat the game one more time.

**DESING:**

1. Ask 1st user for a word
2. Check if the word is valid. Ill try to write a function. Do something like this.

for(unsigned int i =0; i < word.length(); i++)

{

if (word.at(i) == ' ' || !isalpha(word [i]))

cin.clear(); // reset cin

cin. ignore();// varible declaration to discard bad entry

cout << "\nINVALID ENTRY. Try again. ";

}

1. If the user makes valid entry, store it in the variable and ask the 2nd user to guess the game.
2. Declare variables: we need to keep track of misses, guesses, exposed letters
3. Exposed letters will be incremented each time the letter is correctly guessed. This is critical.
4. Initialize: guesses = 3; This is a programmers choice. I decided that user can make only 3 INCORRECT guesses. Please note that I am counting only INCORRECT GUESSES.
5. Initialize string, lets call it display. This string will keep track of letter entered by user 2
6. At this point, we will initialize string display = word;
7. Next step will be to replace all charactesr in the string word with ‘-‘. I saw this code in the book

for (int i = 0; i < display.length(); i++)

display [i] = '-';

1. Next step possibly if stament is (guesses > 0) or possibly while(exposed < word.length() && guesses > 0)

. Until exposed count is less than word. Length., continue this loop.

In this loop we need to let the user know how many guesses they have left.

How many missed they already made

Ask the user to enter a letter

Do cin to let the user enter a letter

1. Once the guess if made, we need to figure out if the guess if correct and if the guess is the duplicate ( possibly 2 bool statement (goodguess= false, duplicate = false)
2. For each entry the user 2 makes, we need to loop through the entry: for(int i=0; i < word.length(); i++) and compare the entry to the word: if(response == word[i])
3. For duplicate do something like this:

if (display [i] == word [i]) . Let the user know that entry is duplicate, change bool to false

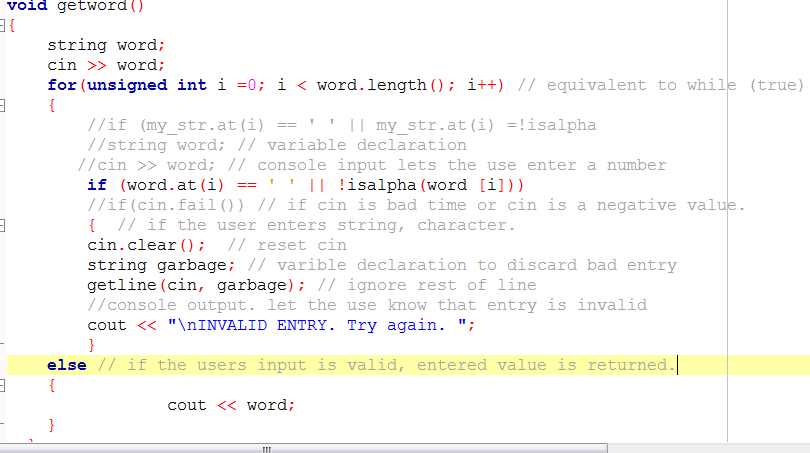
1. If the entry is not duplicate: do else stamen: display[i] = word [i];
   1. Count exposed++;
   2. Set the bool goodGuess =true; // guess was good, so we change bool to good
2. if (!goodGuess)
3. {
4. misses++; // increment misses
5. guesses--;// decrementing guesses
6. // let the user know that the letter is not in the word
7. If there word is guessed : diplay congratulations message
8. If (guesses <=0) the user, ran out of guesses, let the user know that they lost, display the word and let them know which characters they have not guesses. I have no idea at this point how to display the characters that were missed
9. Do while loop to ask the player if they want to play one more time.

**TESTING**

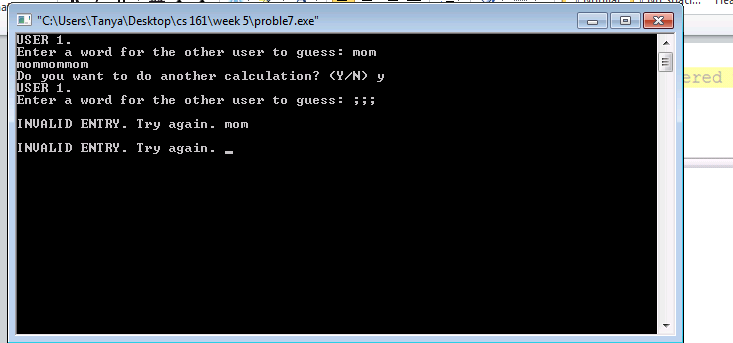
|  |  |  |  |
| --- | --- | --- | --- |
| testing | Input (tested conditions) | Expected output | Pass/Fail |
| User #1 input | Character | Invalid entry, asks to reenter # | pass |
| User # 1 input | number | Invalid entry, asks to reenter # | pass |
| User #1 input | Word has space | Asks user # 2 to guess a number | Pass |
| User # 2 | Guess the letter | The letter replaces ‘-‘ in the word | pass |
| User #2 | Letter is not guesses | User is notified that the letter is not in the word | Pass |
| User #2 | Letter is not guesses | Guesses –  Misses ++ | Pass |
| User # | Letter that was guesses already, entered one more time | Lets the user know that letter was already guesses | Pass |
| Use 2 | Letter that was guesses already, entered one more time | Guesses and misses, stay the same | Pass |
| User 2 | User runs out of guesses and does not guess the word | User is notified that they lost | Pass |
| User 2 | User runs out of guesses and does not guess the word | Secret word if displayed to the user | Pass |
| User 2 | User runs out of guesses and does not guess the word | Letter that were not guessed are displayed to the user | Pass ☺☺☺ |
| User 2 | User guesses the work | Congratulations message is displayed | Pass |
| User 2 | Runs out of guesses | Asks the user if they want to play again. If they say yes, game starts over | Pass |
| User 2 | Runs out of guesses | Asks the user if they want to play again. If they say no, exits the screen | Pass |

**REFLECTION:**

This project was hard. I spent a lot of time on it. My initial problem was that I was not able to design a function that determines if the user 1 entered a word that is valid. First function that I came up with looked somewhat like that:



The problem is represented in the output:



As you can see, if valid word was entered, the program was accepting it, but looping it multiple times depending on how many letters where there in a word. So when word “mom” was entered it was stored as mommommom.

Another problem was that when invalid character was entered first, the program was letting me know that input is invalid. However, when I tried to enter valid word after that, the word was marked as invalid anyways.

I gave up and decided to look at the discussion board. And luckily I came across this;

// function to check that user 1 entered valid word

bool validWord(const string word)

{

for (int i = 0; i < word.length(); i++)// loop through the word

{

// checks if every single character is a letter

if (isalpha(word.at(i)) == 0)

return false;

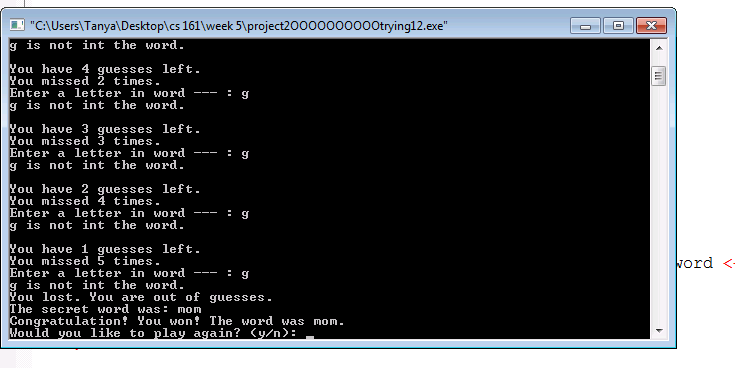
}

return true;

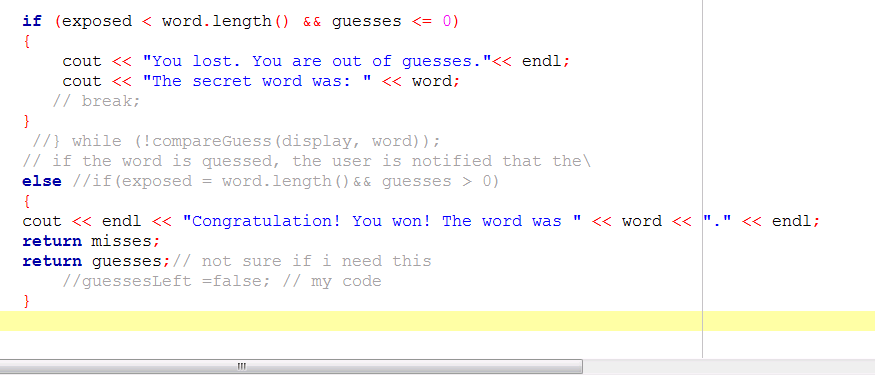
}

This function fixed my problem.

Next major problem that I am still trying to fix and I doubt that Ill be able to is that when user runs out of guesses, for some reason both “congratulation” and “you lost” message are displayed.



If does not make any sense because I am using the following code to determine which message to display:



So if guesses <=0 only a failure message should be displayed. At this point I gave up and started adding more comments. At the end I compiled the program, tested it one more time and it works now. I get only “You lost message” when I run out of guess. I really have no idea how I fixed the problem. The only possible explanation I have is that it was fixed long time ago, but I was just pressing run instead of built/run and it kept running old version with the bug.

Finally I am almost at the end. I still have no idea how to display letter that were not guesses.

Looking at my code, I think that if I try to do something like this:

for (int k=0; k<word.length(); k++){

if (display [k] != word [k])

{

cout << display[k] << ", ";

}}}

This should work. Basically if there is a mismatch between display word and secret word, I should be able to display letters that were not guesses. When I tried to run this, I was getting ‘-‘instead of letters. So instead of cout << display[k] << ", ";,

I decided to cout << word[k] << ", "; and…..

It worked.

I am really sorry for such messy report. Unfortunately, I do not have time to reread it.