Barthelemy Peter

bapeter

CS1300 <AB>

Program 3

num\_Manipulator.cpp

Due Date: 03/22/17

**Source Code**

#include <iostream>

#include <cmath>

#include <cstdlib>

#include <ctime>

using namespace std;

int double\_Num(int temp\_1);

int reversed\_Num(int temp\_2);

int raised\_Num(int temp\_3 , int x);

int combined\_Num(int temp\_4);

int digit\_Count(int temp\_5);

int triple\_Check(int temp\_6);

void prime\_Number(int num);

void step\_By\_Step();

int main()

{

int temp\_1;

int temp\_2;

int temp\_3;

int temp\_4;

int temp\_5;

int temp\_6;

int temp\_7;

int num;

int num\_1;

int x;

srand(time(0));

temp\_1= rand() % 100;

cout <<"\*\*NOTE\*\* Each step is dependent of the value of the previous step\*\*"<< endl<< endl;

cout <<"This is the integer that is being manipulated " <<endl;

cout << temp\_1 << endl << endl;

num = temp\_1;

prime\_Number(num);

step\_By\_Step();

cout << "When '"<< temp\_1 <<"' is doubled, the new number becomes " << endl;

temp\_2 = double\_Num(temp\_1);

cout << temp\_2<< endl << endl;

num = temp\_2;

prime\_Number(num);

step\_By\_Step();

cout << "When '"<< temp\_2 <<"' is reversed, the new number becomes" << endl;

temp\_3 = reversed\_Num(temp\_2);

cout << temp\_3 << endl << endl;

num = temp\_3;

prime\_Number(num);

step\_By\_Step();

cout << "Enter a value (2, 3, or 4) to raise " <<temp\_3<< " to that power " <<endl;

temp\_4 = raised\_Num(temp\_3 , x);

cout << temp\_4<< endl << endl;

num = temp\_4;

prime\_Number(num);

step\_By\_Step();

cout << "This is the sum of the integers of "<<temp\_4<< endl;

temp\_5 = combined\_Num(temp\_4);

cout << temp\_5 << endl << endl;

num = temp\_5;

prime\_Number(num);

step\_By\_Step();

cout << "If the value of '"<<temp\_5<<"' is a two digit number, then the first "

<< endl << "digit will be raised to the power of the second digit"<< endl << endl;

temp\_6 = digit\_Count(temp\_5);

num = temp\_6;

prime\_Number(num);

step\_By\_Step();

cout << "If the value of '"<<temp\_6<<"' is a three digit number and the last digit is less than or equal to 4,"

<< " the last two digit wil be raised to the power of the first digit " << endl << endl;

temp\_7 = triple\_Check(temp\_6);

return 0;

}

int double\_Num(int temp\_1)

{

int num = 2 \* temp\_1;

if (num < 10)

{

cout <<num <<endl << endl;

cout << "\*WAIT\* The value '"<< num <<"' was less than 10, so 10 was automatically added to the value"<< endl;

num += 10;

cout <<"Your new value is "<< num << endl;

}

return num;

}

int raised\_Num(int temp\_3, int x)

{

bool isFound;

isFound = true;

int val;

while(isFound)

{

cin >> x;

if (x >= 2 && x <= 4 )

{

cout << endl;

cout << temp\_3 << " raised to the power of " << x << " = ";

cout << pow(temp\_3,x) << endl << endl;

isFound = false;

val = pow(temp\_3,x);

}

else

{

isFound = false;

cout <<"The value entered is not valid, use only numbers 2, 3, or 5" <<endl << endl;

cout << x << "THIS THE VALUE" << endl;

}

}

if (val < 10)

{

cout << "\*WAIT\* The +value '"<< val <<"' was less than 10, so 10 was automatically added to the value"<< endl;

val += 10;

cout << "Your new value is " << val << endl;

}

return val;

}

int reversed\_Num(int temp\_2)

{

int new\_Num = 0;

while(temp\_2 > 0)

{

new\_Num = new\_Num\*10 + (temp\_2 % 10);

temp\_2 = temp\_2/10;

}

if (new\_Num < 10)

{

cout << new\_Num << endl << endl;

cout << "\*WAIT\* The value '"<< new\_Num <<"' was less than 10, so 10 was automatically added to the value"<< endl;

new\_Num += 10;

cout <<"Your new value is "<< new\_Num << endl;

}

return new\_Num;

}

int combined\_Num(int temp\_4)

{

int sum = 0;

while(temp\_4 > 0)

{

sum += temp\_4 % 10;

temp\_4 /= 10;

}

if (sum < 10)

{

cout << sum << endl << endl;

cout << "\*WAIT\* The value '"<< sum <<"' was less than 10, so 10 was automatically added to the value"<< endl;

sum += 10;

cout <<"Your new value is "<< sum << endl;

}

return sum;

}

int digit\_Count(int temp\_5)

{

int digits;

int x;

int y;

int val ;

int num;

num = temp\_5;

while(temp\_5 != 0)

{

temp\_5 /= 10;

digits++;

}

if (digits == 2)

{

x = num/10 % 10;

y = num % 10;

val = pow(x,y);

cout << "When " << x << " is raised to the power of " << y <<" the new value = " << val << endl <<endl;

}

else

cout << "the number in the previous step is not a two digit number" << endl;

if (val < 10)

{

out << val << endl <<endl;

cout << "\*WAIT\* The value '"<< val <<"' was less than 10, so 10 was automatically added to the value"<< endl;

val += 10;

cout <<"Your new value is "<< val << endl<< endl ;

}

return val;

}

int triple\_Check(int temp\_6)

{

int num\_2;

int digits\_2;

int x;

int y;

int z;

int val;

int num;

num\_2 = temp\_6;

while (temp\_6 != 0)

{

temp\_6 /= 10;

digits\_2++;

}

x = num\_2/100 % 10;

y = num\_2/10 % 10;

z = num\_2 % 10;

if (digits\_2 == 3 && z <= 4)

{

cout << x <<" raised to the power of "<< z << " = " << pow(x,z) <<endl;

num = pow(x,z);

prime\_Number(num);

cout << y <<" raised to the power of " << z << " = " << pow(y,z) <<endl;

num = pow(y,z);

prime\_Number(num);

}

else if (digits\_2 != 3)

cout << "The number is not a three digit number "<< endl;

else if (z > 4)

cout << "The number contains three digits, but the last digit is not less than or equal to 4" <<endl;

return 0;

}

void step\_By\_Step()

{

char confirm;

bool goGet = true;

while(goGet)

{

cout <<"Do you want to continue running this program?"<<endl<<"Y or N"<< endl;

cin >> confirm;

cout<<endl;

{

if(confirm == 'Y')

goGet = false;

else if (confirm =='N')

cout<<endl;

cout << "Click on the 'X' to the top right corner of the window"<< endl;

else

cout << "Invalid input"<< endl;

}

}

}

void prime\_Number(int num)

{

int n=1;

for(int x =2; x < num - 1; x++)

{

if(num%x != 0)

{

}

else

{

n=0;

}

}

if (n==0)

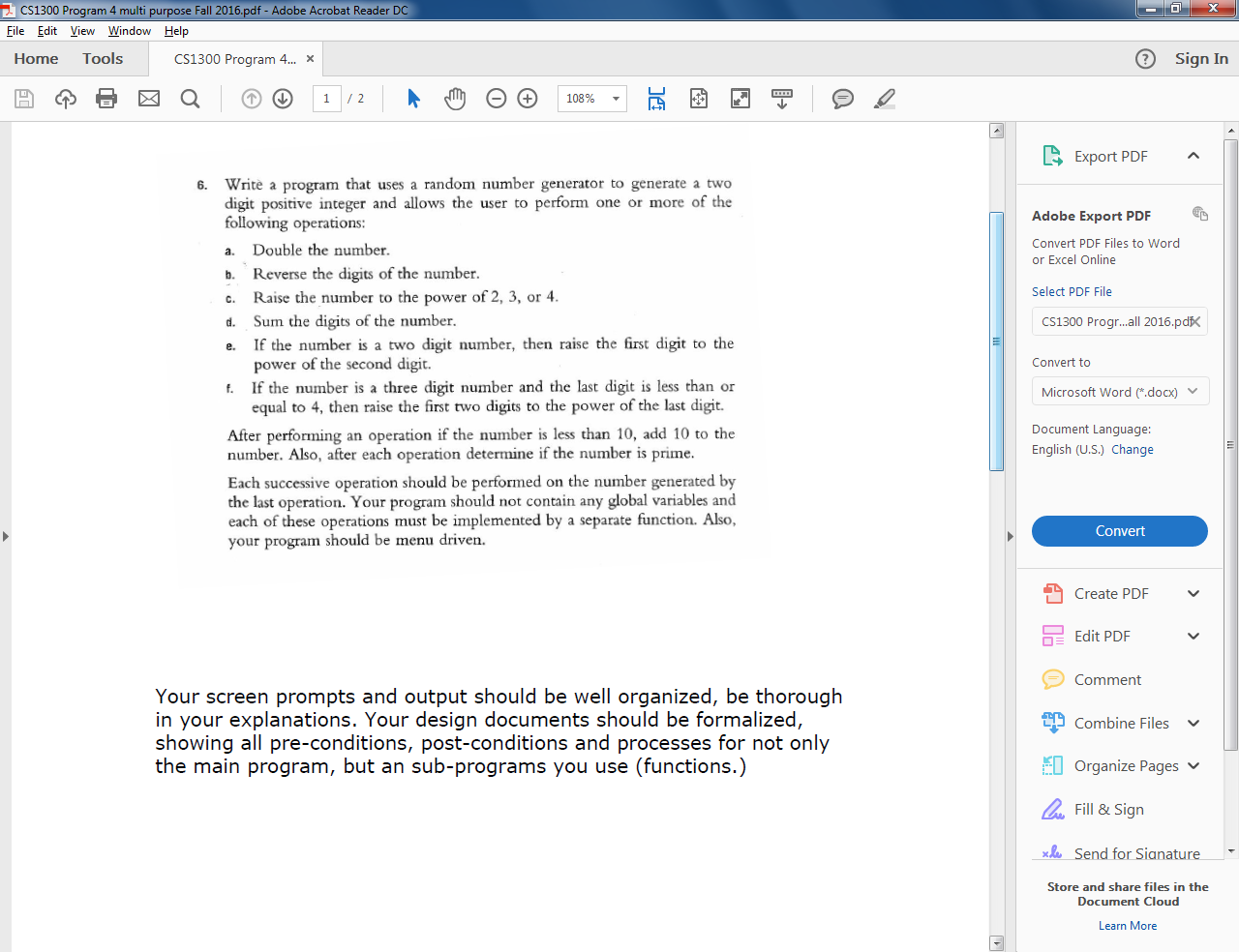
cout <<"'"<< num <<"' is not prime" << endl << endl;

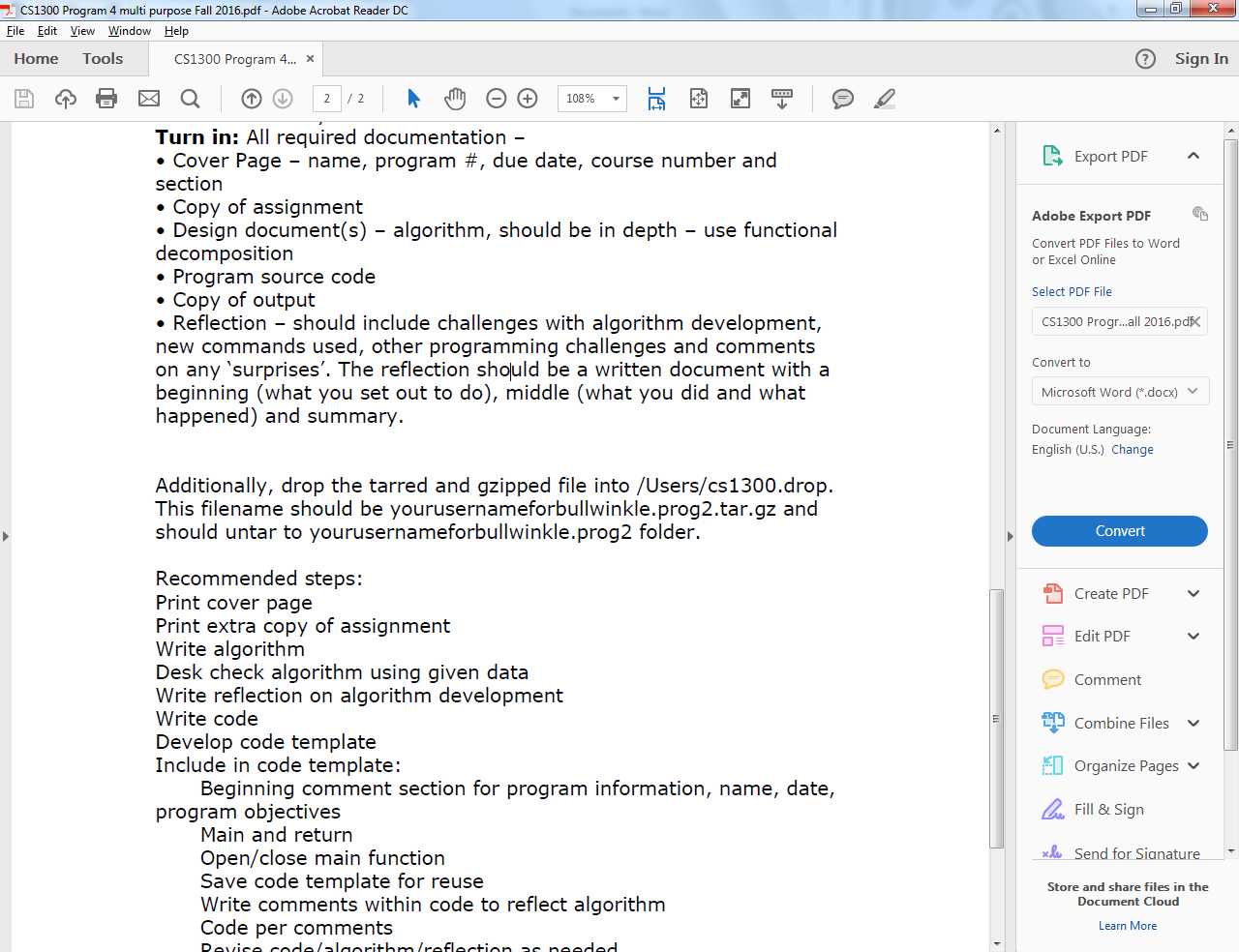
else

cout <<"'"<< num <<"' Number is prime"<< endl << endl;

}

CS1300 Program 3 Due March 22, 2017 Spring 17  
Functions, switch and other C++ code Mansfield





Due March 22, 2017

Design Documents

Design and implement a program that manipulates random numbers and is semi-interactive with the user. The program should function in menu format, giving the user options as the follow the flow of the program. Be sure to make each operation dependent on the operations of the previous step.

**Calculations**

* Finding digits in a number

-number%10…. finds the last digit of the number

-number/10….. removes the last digit from the number

-place in a while loop

* Sum of the digits of a number

-set a variable equal to 0

-variable = variable + number%10

-number/10

-place in a while loop

**Required Input**

* Random number
* User input to raise number to power of (2, 3, or 4)
* Menu: confirm if user would like the continue using the program

**Required output**

* Number doubled
* Reverse of number
* Number raised to the power of (2, 3, or 4)
* Sum of the digits of the number
* Raise first digit to the power of the second digit if the number from the previous step is a two digit number
* Raise the first and second digit to the power of the third digit if the number from the previous step is a three digit number and the last digit is less than or equal to four
* Confirm if the number generated prime or not
* If an operation generates a value less than 10, add 10 to that number

**Pre-conditions**

* User input is minimal
* Initial number is generated at random
* Program executes primarily using functions
* Each operation is dependent on the value of the previous operation
* Global variables are not allowed
* Values being processed are numbers greater or equal to 0 and less than 100

**Post-conditions:**

* All operations of the program work
* Data is well organized
* Output is accurate

**Process**

The Main function will consist mainly of comments outputted to the console which detail each step as the program executes. The main function will also be the means of organizing the overall program in terms of spacing, clarity and readability. Lastly the main function will use function calls will be assigned to variables from simplicity towards the programmer reading the code, numbered in the order of when the process is occurring.

Sub-functions are the most important part of this program because of the reusability of the code. Int functions will dominate most of the functions since non-decimal values are being read. Few void functions will be used to slip in repetitive lines of code where necessary.

**Output Listing**

\*\*NOTE\*\* Each step is dependent of the value of the previous step\*\*

This is the integer that is being manipulated

25

'25' is not prime

Do you want to continue running this program?

Y or N

Y

When '25' is doubled, the new number becomes

50

'50' is not prime

Do you want to continue running this program?

Y or N

Y

When '50' is reversed, the new number becomes

5

\*WAIT\* The value '5' was less than 10, so 10 was automatically added to the value

Your new value is 15

15

'15' is not prime

Do you want to continue running this program?

Y or N

Y

Enter a value (2, 3, or 4) to raise 15 to that power

4

15 raised to the power of 4 = 50625

50625

'50625' is not prime

Do you want to continue running this program?

Y or N

Y

This is the sum of the integers of 50625

18

'18' is not prime

Do you want to continue running this program?

Y or N

Y

If the value of '18' is a two digit number, then the first

digit will be raised to the power of the second digit

When 1 is raised to the power of 8 the new value = 1

1

\*WAIT\* The value '1' was less than 10, so 10 was automatically added to the value

Your new value is 11

'11' Number is prime

Do you want to continue running this program?

Y or N

Y

If the value of '11' is a three digit number and the last digit is less than or equal to 4, the last two digit will be raised to the power of the first digit

The number is not a three digit number

**Reflection**

**Overview:**

The purpose of this lab was to learn how to manipulate integers. Through manipulating these integers the programmer also learns to not be dependent on global variables and how to manipulate user defined functions to make a tidier, more efficient running program. Lastly the programmer also learned how to create a menu of options that the user could use to interact with the running program.

**Challenges and Solution:**

I encountered several challenges with this program such as calculating the number of digits in a number, checking if the number was prime, and error checking. Finding digits was resolved by research and application of concepts learned from logical operators; similar, prime number checking was conquered due to perseverance and application of concepts. Error checking was the most difficult challenge because some errors still remain unresolved. For example, I tried to make the char value of ‘y’ equal to the char value of ‘Y’ and likewise with ‘n’ but my program was program would not run as expected which in turn lead to my creation of a case sensitive menu. Also, when generating user input for a integer to be raised to a certain power I ran into an issue where if and invalid input were not a number value my program would spit out an infinite loop.

**Lessons Learned:**

Through this program, I learned how to use different types of while loops and also to better understand the value of user defined functions within a program. I also learned more about the mod operator shortcut operators which can be used to tidy up sloppy code. Lastly the most important thing I learned from this program was the importance of understanding algebra and the ability to follow lines of code.