EGR 125 - Introduction to Engineering Methods (C++)

File: N125-Ch6LB (kg spring2017)

Due date: \_\_\_10/14/2021\_\_

**Chapter 6 Homework(b)**

**Reading Assignment:**

Read Chapter 6 in Introduction to Programming with C++, 3rd Edition by Liang

1. **Checkpoint Exercises**: Work the following problems in the textbook. Work each problem by hand (not using thecompiler). ***Be sure to write out the instructions for each problem and include the given information.***
   1. (7 pts) Checkpoint **Exercise 6.8**: Write function headers (not the bodies) for the following functions:
2. Return a sales commission, given the sales amount and the commission rate:

***double salecom (double, double);***

1. Display the calendar for a month, given the month and year:

***void calendar (int, int);***

1. Return a square root of a number:

***double sqrt(double);***

1. Test whether a number is even, and return true if it is:

***bool testEven (int);***

1. Display a message a specified number of times:

***void printMessage (int);***

f. Return the monthly payment, given the loan amount, number of years, and annual

interest rate:

***double paymentMonthly (double, double, double);***

g. Return the corresponding uppercase letter, given a lowercase letter:

***char lowercase (char);***

* 1. (6 pts) Checkpoint **Exercise 6.18**: Show the output of the following code:

#include <iostream>

using namespace std;

const double PI = 3.14159;

double getArea(double radius)

{

return radius \* radius \* PI;

}

void displayArea(double radius)

{

cout << getArea(radius) << endl;

}

int main()

{

double r1 = 1;

double r2 = 10;

cout << getArea(r1) << endl;

displayArea(r2);

}

***3.14159***

***314.159***

**Programming Exercises:**

In each case turn in a printout of the main program, any functions, and the output for any test cases indicated. In general, you don’t need to protect against illegal inputs for homework problems unless specifically noted. Use the standard TCC template. Also include comments at the beginning of all functions describing the function. Always include a prototype with each function and show the function after the main function, not before it. NOTE: flowcharts are not required.

1. (15 pts) Function with one return value: Write a C++ function that returns volume of a sphere in cubic meters given the radius in meters. Write a main program that calls the function within a loop in order to print the radius and volume in a table as the radius varies from 1 m to 10 m. The table might appear as shown below:

|  |  |
| --- | --- |
| Sphere | Sphere |
| Radius (m) | Volume (m^3) |
| 1 | 4.19 |
| 2 |  |
| .. |  |
| 10 |  |

1. (15 pts) Functions with one return value: Write a C++ program for Programming **Exercise 6.9** (*Conversions between* *Celsius and Fahrenheit*) on page 254.

The table produced should be shown exactly as in the text (use left justification for each column, use the number of digits after the decimal point shown in each column, show trailing zeros, etc)

1. (15 pts) Function with no return values:

Write a C++ program for Programming **Exercise 6.11**: Write a function that prints the ASCII values of the characters using the following header: void printASCII (char ch1, char ch2, int numberPerLine)

This function prints the ASCII values of characters between ch1 and ch2. Test the program for the following cases:

* 1. All upper case letters using 6 characters per line
  2. All lower case letters using 8 characters per line
  3. All characters from ‘1’ to ‘Z’ using 10 characters per line

1. (15 points) Reference Parameters (returning multiple values): Write a C++ function that converts standard time to military time. Inputs include hours and minutes in standard time and a character equal to ‘a’ for am or ‘p’ for pm.

The function call might look like:

**MilitaryTime(SHour, SMin, AorP, MHour, MMin);**

Also write a main program to prompt the user for the inputs (such as 1:30 am), call the function, and display the input and the output in the following form:

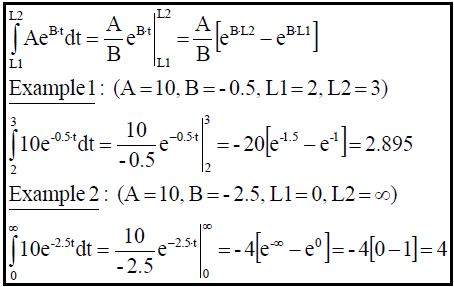
* + - 12:30 am = 0030
    - 2:30 am = 0230
    - 3:30 pm = 1530
    - 12:00 am = 0000

Run the program for the four cases above plus at least three other cases.

Hint: Using fill(’0’) is an easy way to show leading zeros.

5. (15 pts) Default arguments: Write a C++ function **ExpoArea** that will return the area under an exponential function of the form AeBt by integrating the function from limits L1 to L2.

Recall that:



Hint: You don’t need to integrate. Just use this result:

The function should have inputs for A, B, L1, and L2.

|  |
| --- |
|  |

NOTE: L1 and L2 should have ***default arguments*** of 0 and 1E100 (approximately ∞). Since L1 and L2 are default arguments, they can be omitted if L1 = 0 and L2 = ∞. (If you don’t recall what ***default arguments*** are, find an example in the class notes before proceeding.)

***Examples:***

Result = **ExpoArea** (10, -0.5, 2,3) should return a value of 2.895 Result = **ExpoArea** (10, -2.5) should return a value of 4

1. Write function **ExpoArea** as described above.
2. Write a main function that calls the function 5 times (once for each of the 5 test cases below) and displays the area. Use the minimum number of inputs required with the function in each case. No user inputs are required.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case # | A | B | L1 | L2 |
| 1 | 10 | -0.5 | 2 | 3 |
| 2 | 10 | -2.5 | 0 | ∞ |
| 3 | 100 | -20 | 0 | 0.1 |
| 4 | 500 | -100 | 0.005 | ∞ |
| 5 | 500 | -100 | 0 | ∞ |

6. (12 pts) Determine the output for each part below by hand (do not compile the programs). You can simply fill out and turn in this page.

|  |  |
| --- | --- |
| // Part 1 - FunctionTest1B | |
| #include <iostream>  #include <cmath>  using namespace std;  double Times2(double); // Function declarations  double Times3(double);  int main ( void ) // Main function  {double x = 4.0, y = 6.0,z;  z = Times2(x);  x = Times3(x);  y = Times3(Times2(x) + Times2(y));  cout << "x = " << x << endl;  cout << "y = " << y << endl;  cout << "z = " << z << endl;  return 0;}  double Times2(double x) //Function definitions  { double y = 2\*x;  return y;}  double Times3(double y)  { double x = 3\*y;  return x;} | x = ***\_\_\_\_\_\_12\_\_\_\_\_\_***  y = ***\_\_\_\_108\_\_\_\_\_\_\_\_***  z = ***\_\_\_8\_\_\_\_\_\_\_\_\_*** |
| // Part 2 - FunctionTest2B | |
| #include <iostream>  #include <cmath>  using namespace std;  // Function declaration  void F2B(int,int,int,int&,int&,int&);  // Main function  int main ( void )  { int X=2,Y=3,Z=4,A=5,B=6,C=7,X1=8,Y1=9,Z1=10;  F2B(A,B,C,X,Y,Z); // function call  Y+=2;  B+=4;  Y1 += 6;  cout << "A = " << A << "\nB = " << B << "\nC = " << C << endl;  cout << "X = " << X << "\nY = " << Y << "\nZ = " << Z << endl;  cout << "X1 = " << X1 << "\nY1 = " << Y1 << "\nZ1 = " << Z1 << endl;  return 0; }  // Function definition  void F2B(int A, int B, int C, int& X1, int& X2, int& X3)  { A+=2;  X1 += 4;  X2 = (A + B)/2;  X3 = (A - B)/2;  return;} | A = **5**  B = **10**  C = **7**  X = **6**  Y = **8**  Z = **0**  X1 = **8**  Y1 = **15**  Z1 = **10** |

**Program 1:**

// \_\_\_\_ \_\_ \_\_

// / / \ / \

// / / /

// / \\_\_\_/ \\_\_\_/

//

// Date: 10/13/2021

// Name: David Vermaak

// Project Description: This program outputs a formatted list using a function that

// calculates the volume of a sphere

// Inputs: n/a

// Outputs: a formatted list

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <iostream> //This header containing cout and cin

#include <conio.h> //This header declares getch which pauses until a key is pushed

#include <math.h> //This header allows for the use of more complex mathematical operators

#include <string> //This header enables string functions

#include <iomanip> //This header contains functions to better format output

using namespace std; //introduces namespace std

const double pi = acos(-1); //constant for pi

double VolumeSp (double); //Function declaration

int main ( )

{

cout << left << setw (12) << "Sphere" << setw (12) << "Sphere" << endl;

cout << left << setw (12) << "Radius (m)" << setw (12) << "Volume (m^3)" << endl;

for (int i=1; i<=10;i++)

cout << left << setw (12) << i << setw (12) << VolumeSp (i) << endl;

return 0;

}

//Function definition

double VolumeSp (double r) //calculates the volume of a sphere using radius

{

return (pow(r,3) \* (4.0/3.0) \* pi);

}

**Program 1 Output:**

Sphere Sphere

Radius (m) Volume (m^3)

1 4.18879

2 33.5103

3 113.097

4 268.083

5 523.599

6 904.779

7 1436.76

8 2144.66

9 3053.63

10 4188.79

**Program 2:**

// \_\_\_\_ \_\_ \_\_

// / / \ / \

// / / /

// / \\_\_\_/ \\_\_\_/

//

// Date: 10/13/2021

// Name: David Vermaak

// Project Description: Converts between Celsius and Fahrenheit using functions

// Inputs: n/a

// Outputs: a formatted table

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <iostream> //This header containing cout and cin

#include <iomanip> //This header contains functions to better format output

using namespace std; //introduces namespace std

double F2C (double); //Function declaration

double C2F (double);

int main ( )

{

cout << left << setw (12) << "Fahrenheit" << setw (12) << "Celsius\t| "

<< left << setw (12) << "Celsius" << setw (12) << "Fahrenheit" << endl;

for (int i=0; i<=100; i= i + 10)

cout << left << setw (12) << i << setw (12) << F2C (i) << "| "

<< left << setw (12) << i << setw (12) << C2F (i)<< endl;

return 0;

}

//Function definition

double F2C (double f) //calculates Celsius from Fahrenheit

{

return ((f- 32) \* (5.0/9.0));

}

//Function definition

double C2F (double c) //calculates Fahrenheit from Celsius

{

return (c\*(9.0/5.0) + 32);

}

**Program 2 Output:**

Fahrenheit Celsius | Celsius Fahrenheit

0 -17.77 | 0 32

10 -12.22 | 10 50

20 -6.66 | 20 68

30 -1.11 | 30 86

40 4.44 | 40 104

50 10 | 50 122

60 15.55 | 60 140

70 21.11 | 70 158

80 26.66 | 80 176

90 32.22 | 90 194

100 37.77 | 100 212

**Program 3:**

// \_\_\_\_ \_\_ \_\_

// / / \ / \

// / / /

// / \\_\_\_/ \\_\_\_/

//

// Date: 10/13/2021

// Name: David Vermaak

// Project Description: Program prints the characters between (c1,c2) with n per line

// Inputs: two chars and an int

// Outputs: a list of chars

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <iostream> //This header containing cout and cin

#include <conio.h> //This header declares getch which pauses until a key is pushed

#include <math.h> //This header allows for the use of more complex mathematical operators

#include <string> //This header enables string functions

#include <iomanip> //This header contains functions to better format output

using namespace std; //introduces namespace std

void printASCII (char ch1, char ch2, int numberPerLine); //Function declaration

int i=0;

int main ( )

{

printASCII('A','Z',6);

return 0;

}

//Function definition

void printASCII (char c1, char c2, int n) //outputs the characters between (c1,c2) with n per line

{

while (c1<=c2)

{cout << c1 << " ";

c1++;

i++;

if (n<=i)

{

cout <<endl;

i=0;

}

}

return;

}

**Program 3 Output:**

|  |  |  |
| --- | --- | --- |
|  |  |  |

**Program 4:**

// \_\_\_\_ \_\_ \_\_

// / / \ / \

// / / /

// / \\_\_\_/ \\_\_\_/

//

// Date: 10/13/2021

// Name: David Vermaak

// Project Description: Template

// Inputs: hours and minutes in standard time and a character to denote am or pm

// Outputs: standard time and the equivalent military time

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <iostream> //This header containing cout and cin

#include <string> //This header enables string functions

#include <iomanip> //This header contains functions to better format output

using namespace std; //introduces namespace std

void MilitaryTime(int Hour, int Min, char AMorPM); //Function declaration

int hr, Amin;

char am\_pm;

int main ( )

{

cout << "Please enter the time int the form Hour (enter) Minute (enter)\n";

cin >> hr;

cin >> Amin;

cout << "Please denote AM or PM by entering A or P, respectively\n";

cin >> am\_pm;

cout << "\n\n\n\n\nStandard Time\t Military Time\n" << hr << ":" << Amin << " "<< am\_pm << "M \t ";

MilitaryTime(hr, Amin, am\_pm);

return 0;

}

//Function definition

void MilitaryTime(int H, int M, char A\_P) //changes standard time to military time

{

if (A\_P == 'A' || A\_P == 'a')

{

if (H >= 12)

{cout << "00" << M;}

else

cout << "0"<< H << M;

}

else if (A\_P == 'P' || A\_P == 'p')

{cout << H+12 << M;}

return;

}

**Program 4 Output:**

Please enter the time int the form Hour (enter) Minute (enter)

12

30

Please denote AM or PM by entering A or P, respectively

A

Standard Time Military Time

12:30 AM 0030

Standard Time Military Time

2:30 AM 0230

Standard Time Military Time

3:30 PM 1530

Standard Time Military Time

12:00 AM 0000

Standard Time Military Time

5:15 PM 1715

Standard Time Military Time

4:59 PM 1659

Standard Time Military Time

11:36 AM 1136

**Program 5:**

// \_\_\_\_ \_\_ \_\_

// / / \ / \

// / / /

// / \\_\_\_/ \\_\_\_/

//

// Date: 10/13/2021

// Name: David Vermaak

// Project Description: Calculates the area under the equation from Limit 1 to Limit 2

// Inputs: A, B, L1, L2

// Outputs: Area

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <iostream> //This header containing cout and cin

using namespace std; //introduces namespace std

double ExpoArea (double A, double B, double Limit1 = 0, double Limit2 = 1e100 ); //Function declaration

double x, A1, B1, l1, l2;

int main ( )

{

cout << "1 " << ExpoArea(10, -0.5, 2, 3);

cout << "\n2 " << ExpoArea(10, -2.5);

cout << "\n3 " << ExpoArea(100, -20, 0, 0.1);

cout << "\n4 " << ExpoArea(500, -100, 0.005);

cout << "\n5 " << ExpoArea(500, -100);

return 0;

}

double ExpoArea (double a, double b, double L1, double L2) //Calculates the area under the function from L1 to L2

{

return ((a/b)\*(exp(b\*L2)-exp(b\*L1)));

}

**Program 5 Output:**

|  |  |
| --- | --- |
| Test case # | Area |
| 1 | 2.89499 |
| 2 | 4 |
| 3 | 4.32332 |
| 4 | 3.03265 |
| 5 | 5 |