

**KARAKORAM INTERNATIONAL UNIVERSITY**

**Name: Haffiz-Ud-Din**

**Reg No: 2023-KIU-BS4141**

**Teacher: Mr. Sajjid Hussain**

**Subject: Fundamentals of Programming**

**Assignment No: 1**

1. Add one or more of your own unique functions to the stick figure program presented in class Create several more pictures combining the drawCircle, drawIntersect, drawBase, and drawParallel functions with your own. Make any modifications to these functions that you need in order to make the picture components fit nicely.

#include <iostream>

using namespace std;

    void drawCircle()

   {

        cout << "  \*\*\*\*  \n";

      cout << "\*      \*\n";

      cout << "\*      \*\n";

    cout << "  \*\*\*\*  \n";

 }

    void drawRectangle()

  {

        cout << "\*\*\*\*\*\*\*\*\*\n";

        cout << "\*       \*\n";

        cout << "\*       \*\n";

        cout << "\*\*\*\*\*\*\*\*\*\n";

}

    void drawParallelLines()

  {

        cout << "  ||||  \n";

}

     void drawTriangle()

    {

        cout<<"   /\\\n";

        cout<<"  /  \\\n";

        cout<<" /    \\\n";

        cout<<"/      \\\n";

}

     void drawBaseLine()

  {

         cout << "\_\_\_\_\_\_\_\_\_\n";

}

     void drawBalloon()

  {

         cout << "   o   \n";

         cout << "  /|\\  \n";

         cout << "   |   \n";

}

     int main()

{

         cout << "Stick Figure:\n\n";

         drawCircle();

         drawTriangle();

         drawBaseLine();

         drawParallelLines();

         drawParallelLines();

         drawParallelLines();

         cout <<"\n"<< "Stick Figure With Balloon:\n\n";

         drawBalloon();

         drawCircle();

         drawTriangle();

         drawBaseLine();

         drawParallelLines();

         drawParallelLines();

         drawParallelLines();

    return 0;

}

1. Write three functions, one that displays a circle, one that displays a rectangle, and one that displays a triangle. Use these functions to write a complete C++ program from the following outline: int main()

{

// Draw circle.

// Draw triangle.

// Draw rectangle.

// Display 2 blank lines.

// Draw triangle.

// Draw circle.

// Draw rectangle.

return 0;

}

#include <iostream>

using namespace std;

void drawCircle()

{

    cout << "  \*\*\*\*\*  \n";

    cout << "\*       \*\n";

    cout << "\*       \*\n";

    cout << "  \*\*\*\*\*  \n";

}

void drawRectangle() {

    cout << "\*\*\*\*\*\*\*\*\*\n";

    cout << "\*       \*\n";

    cout << "\*       \*\n";

    cout << "\*\*\*\*\*\*\*\*\*\n";

}

void drawTriangle() {

    cout << "    \*    \n";

    cout << "   \*\*\*   \n";

    cout << "  \*\*\*\*\*  \n";

}

int main() {

    // Draw circle

    drawCircle();

    // Draw triangle

    drawTriangle();

    // Draw rectangle

    drawRectangle();

    // Display 2 blank lines

    cout << "\n\n";

    // Draw triangle

    drawTriangle();

    // Draw circle

    drawCircle();

    // Draw rectangle

    drawRectangle();

    return 0;

}

1. Write a computer program that computes the duration of a projectile’s flight and its height above the ground when it reaches the target. As part of your solution, write and call a function that displays instructions to the program user.

**Problem Constant**

G = 32.17 // gravitational constant

**Problem Inputs**

float theta // input - angle (radians) of elevation

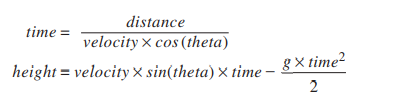
float distance // input - distance (ft) to target

float velocity // input - projectile velocity (ft/sec)

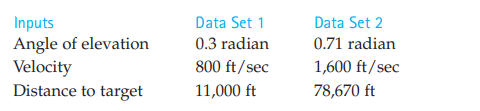
**Problem Outputs**

float time // output - time (sec) of flight

float height // output - height at impact

 Relevant Formulas

Try your program on these data sets.



#include <iostream>

#include <cmath>

using namespace std;

void displayInstructions()

{

    cout << "Projectile Motion Calculator\n";

    cout << "Enter the following information:\n";

    cout << "1. Angle of elevation (in radians): ";

}

int main()

{

  const float G = 32.17; // gravitational constant (ft/sec^2)

    float theta, distance, velocity, time, height;

    displayInstructions();

    cin >> theta;

    cout << "2. Distance to target (in feet): ";

    cin >> distance;

    cout << "3. Projectile velocity (in ft/sec): ";

    cin >> velocity;

    // Calculate the time of flight

    time = distance / (velocity \* cos(theta));

    // Calculate the height at impact

    height = velocity \* sin(theta) \* time - (G \* time \* time) / 2;

    cout << "\nDuration of flight: " << time << " seconds\n";

    cout << "Height at impact: " << height << " feet\n";

    return 0;

}

1. Write a program that takes a positive number with a fractional part and rounds it to two decimal places. For example, 32.4851 would round to 32.49, and 32.4431 would round to 32.44.

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

    double number, roundedNumber;

    cout << "Enter a positive number with a fractional part: ";

    cin >> number;

    if (number < 0)

  {

        cout << "Please enter a positive number." << endl;

        return 1; // Exit with an error code

    }

    roundedNumber = round(number \* 100.0) / 100.0;

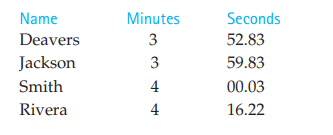
    cout << "Original number: " << number << endl;

    cout << "Rounded to two decimal places: " << roundedNumber << endl;

    return 0;

}

1. Four track stars entered the mile race at the Penn Relays. Write a program that will read the last name and the race time in minutes and seconds for one runner and compute and print the speed in feet per second and in meters per second after the runner’s name. (Hints: There are 5280 feet in one mile, and one kilometer equals 3281 feet; one meter is equal to 3.281 feet.) Test your program on each of the times below



Write and call a function that displays instructions to the program user. Write two other functions, one to compute the speed in meters per second and the other to compute the speed in feet per second.

#include <iostream>

#include <string>

#include <iomanip>

using namespace std;

const double FEET\_PER\_MILE = 5280;

const double FEET\_PER\_KILOMETER = 3281;

const double METERS\_PER\_FOOT = 1.0 / 3.281;

void displayInstructions()

{

    cout << "Enter the last name of the runner and the race time in minutes and seconds:\n";

    cout << "Example: Smith 4 30\n";

}

double computeSpeedInFeetPerSecond(double time)

 {

    return FEET\_PER\_MILE / time;

}

double computeSpeedInMetersPerSecond(double time)

{

    return FEET\_PER\_MILE \* METERS\_PER\_FOOT / time;

}

int main()

{

    string lastName;

    double minutes, seconds;

    displayInstructions();

    cin >> lastName >> minutes >> seconds;

    double totalTimeInSeconds = minutes \* 60 + seconds;

    cout << lastName << ": "<<setw(2);

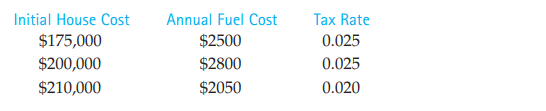
    cout << "Speed in feet/second: " << computeSpeedInFeetPerSecond(totalTimeInSeconds) << " ft/s\n" ;

    cout <<"Speed in meters/second: " << computeSpeedInMetersPerSecond(totalTimeInSeconds) << " m/s\n";

    return 0;

}

1. In shopping for a new house, you must consider several factors. In this problem the initial cost of the house, estimated annual fuel costs, and annual tax rate are available. Write a program that will determine the total cost after a five-year period for each set of house data below. You should be able to inspect your program output to determine the “best buy.”



To calculate the house cost, add the fuel cost for five years to the initial cost, then add the taxes for five years. Taxes for one year are computed by multiplying the tax rate by the initial cost. Write and call a function that displays instructions to the program user and another function that computes and returns the house cost given the initial cost, the annual fuel cost, and the tax rate.

#include <iostream>

#include <iomanip>

using namespace std;

// Function to display instructions

void displayInstructions()

{

    cout << "Enter the following information for each house:\n";

    cout << "1. Initial cost of the house: ";

    cout << "2. Estimated annual fuel costs: ";

    cout << "3. Annual tax rate: ";

}

// Function to compute the total cost after five years

double computeTotalCost(double initialCost, double annualFuelCost, double taxRate) {

    // Calculate taxes for one year

    double annualTaxes = initialCost \* taxRate;

    // Calculate fuel cost for five years

    double totalFuelCost = annualFuelCost \* 5;

    // Calculate total cost after five years

    double totalCost = initialCost + totalFuelCost + (annualTaxes \* 5);

    return totalCost;

}

int main() {

    displayInstructions();

    double initialCost, annualFuelCost, taxRate;

    cin >> initialCost >> annualFuelCost >> taxRate;

    double totalCost = computeTotalCost(initialCost, annualFuelCost, taxRate);

    cout << "Total cost after five years: $" << setw(2) << totalCost << endl;

    return 0;

}

1. Write a program that reads a string containing exactly four words (separated by \* symbols) into a single string object. Next, extract each word from the original string and store each word in a string object. Then concatenate the words in reverse order to form another string. Display both the original and final strings. (Hint: To extract the words, you should use the find member function to find each symbol \*, assign the characters up to the \* to one of the four string objects, and then remove those characters from the original string.)

#include <iostream>

#include <string>

using namespace std;

int main() {

    string inputString;

    string word1, word2, word3, word4;

    string finalString;

    // Read the input string containing exactly four words separated by asterisks

    cout << "Enter a string with four words separated by \* symbols: ";

    getline(cin, inputString);

    // Find and extract the first word

    size\_t firstAsterisk = inputString.find('\*');

    word1 = inputString.substr(0, firstAsterisk);

    inputString.erase(0, firstAsterisk + 1);

    // Repeat the process for the remaining words

    size\_t secondAsterisk = inputString.find('\*');

    word2 = inputString.substr(0, secondAsterisk);

    inputString.erase(0, secondAsterisk + 1);

    size\_t thirdAsterisk = inputString.find('\*');

    word3 = inputString.substr(0, thirdAsterisk);

    inputString.erase(0, thirdAsterisk + 1);

    // The remaining part of the input string is the fourth word

    word4 = inputString;

    // Concatenate the words in reverse order

    finalString = word4 + " " + word3 + " " + word2 + " " + word1;

    // Display the original and final strings

    cout << "Original string: " << word1 << " \* " << word2 << " \* " << word3 << " \* " << word4 << endl;

    cout << "Final string (reversed): " << finalString << endl;

    return 0;

}

1. Write a program to take a depth (in kilometers) inside the earth as input data; compute and display the temperature at this depth in degrees Celsius and Fahrenheit. The relevant formulas are

Celsius = 10 × (depth) + 20 (Celsius temperature at depth in km)

Fahrenheit = 1.8 × (Celsius) + 32

Include two functions in your program. Function celsiusAtDepth should compute and return the Celsius temperature at a depth measured in kilometers. Function toFahrenheit should convert a Celsius temperature to Fahrenheit.

#include <iostream>

using namespace std;

double celsiusAtDepth(double depth)

{

    return 10.0 \* depth + 20.0;

}

double toFahrenheit(double celsius)

{

    return 1.8 \* celsius + 32.0;

}

int main()

{

    double depth;

    cout << "Enter the depth inside the Earth (in kilometers): ";

    cin >> depth;

    double celsius = celsiusAtDepth(depth);

    double fahrenheit = toFahrenheit(celsius);

    cout << "Temperature at " << depth << " kilometers: " << celsius << " degrees Celsius\n";

    cout << "Temperature at " << depth << " kilometers: " << fahrenheit << " degrees Fahrenheit\n";

    return 0;

}

1. After studying the population growth of Gotham City in the last decade of the twentieth century, we have modeled Gotham’s population function as

P(t) = 52.966 + 2.184t

where t is years after 1990, and P is population in thousands. Thus, P(0) represents the population in 1990, which was 52.966 thousand people. Write a program that defines a function named population that predicts Gotham’s population in the year provided as an input argument. Write a program that calls the function and interacts with the user as follows:

Enter a year after 1990> 2015

Predicted Gotham City population for 2010 (in thousands):

107.566

#include <iostream>

using namespace std;

double population(int year)

{

    // Population function P(t) = 52.966 + 2.184t (in thousands)

    double population = 52.966 + 2.184 \* (year - 1990);

    return population;

}

int main()

{

    int year;

    cout << "Enter a year after 1990: ";

    cin >> year;

    if (year >= 1990)

  {

        double predictedPopulation = population(year);

        cout << "Predicted Gotham City population for " << year << " (in thousands): " << predictedPopulation << endl;

    } else

  {

        cout << "Please enter a year after 1990." << endl;

    }

    return 0;

}

1. Write a program that calculates and displays the volume of a box and the surface area of a box. The box dimensions are provided as input data. The volume is equal to the area of the base times the height of the box. Define and call a function rectangleArea to calculate the area of each rectangle in the box.

#include <iostream>

using namespace std;

// Function to calculate the area of a rectangle

double rectangleArea(double length, double width)

{

    return length \* width;

}

int main()

{

    double length, width, height;

    // Input box dimensions

    cout << "Enter the length of the box: ";

    cin >> length;

    cout << "Enter the width of the box: ";

    cin >> width;

    cout << "Enter the height of the box: ";

    cin >> height;

    // Calculate the volume of the box (area of base \* height)

    double volume = rectangleArea(length, width) \* height;

    // Calculate the surface area of the box (2 \* area of each rectangle + 2 \* area of each rectangle + 2 \* area of each rectangle)

    double surfaceArea = 2 \* (rectangleArea(length, width) + rectangleArea(width, height) + rectangleArea(height, length));

    cout << "Volume of the box: " << volume << endl;

    cout << "Surface area of the box: " << surfaceArea << endl;

    return 0;

}