// comsc 200

// main.cpp

// 8a

// completed

// Created by Jeff on 10/3/16.

// Copyright © 2016 Jeff zhang. All rights reserved.

//

#include <iostream>

#include <iomanip>

#include <sstream>

using namespace std;

class Shap{

protected:

double area;

public:

virtual void setArea() = 0;

};

class Rectangle:public Shap{

private:

long width,length;

public:

Rectangle(long w,long l){

width = w;

length = l;

setArea();

}

double getArea()const {

return area;

};

void setArea(){

area= width\*length;

}

};

class Circle:public Shap{

private:

long centerx;

long centery;

double radius;

public:

Circle(long x,long y, long r){

centerx=x;

centery = y;

radius = r;

setArea();

}

double getArea()const{

return area;

}

void setArea(){

area= 3.14\*radius\*radius;

}

};

int main(){

// instanciate abstract class test

// Shape s();

// Demonstrate a Circle.

long x,y,length,width;

double rad;

std::cout << "Enter the x, y coordinate and radius of the circle's center: ";

std::cin >> x >> y >> rad;

std::cout << "\nCordinate x: " << x << ", y: " << y << ", and radius: " << rad << "\n";

Circle c(x,y,rad);

std::cout << "The area of the circle is " << c.getArea() << ".";

// Demonstrate a Rectangle.

std::cout << "\n\nPlease enter the length and width of the rectangle: ";

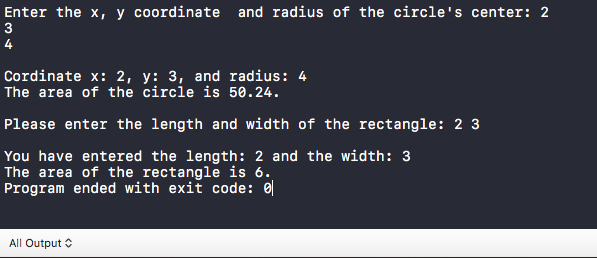
std::cin >> length >> width;

std::cout << "\nYou have entered the length: " << length << " and the width: " << width << "\n";

Rectangle r(width, length);

std::cout << "The area of the rectangle is " << r.getArea() << ".\n";

}



// comsc 200

// main.cpp

// 8b

// completed

// Created by Jeff on 10/3/16.

// Copyright © 2016 Jeff zhang. All rights reserved.

//

#include <iostream>

#include "myName.h"

#include <iomanip>

#include <cmath>

using namespace std;

int main() {

cout << "namespace std \n"

<<"7 ^ 3 = " << pow(7,3) << endl;

cout << "7.1 ^ 3.1 = " << pow(7.1, 3.1) <<endl;

cout << "namespace mymath1 \n"

<<"7 ^ 3 = " << mymath1::pow(7,3) << endl;

cout << "7.1 ^ 3.1 = " << mymath1::pow(7.1, 3.1) <<endl;

cout << "namespace mymath1 \n"

<<"7 ^ 3 = " << mymath2::pow(7,3) << endl;

cout << "7.1 ^ 3.1 = " << mymath2::pow(7.1, 3.1) <<endl;

}

//

// myName.h

// 8b

//

// Created by Jeff on 10/3/16.

// Copyright © 2016 Jeff zhang. All rights reserved.

//

#ifndef myName\_h

#define myName\_h

namespace mymath1{

double pow(double a, int b){

double c=1.0;

for(b; b>0; b--){c\*=a;}

return c;}}

namespace mymath2{

double pow(double a, int b){

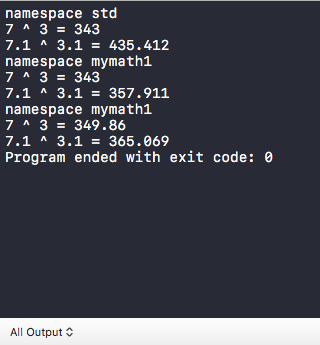
double c=1.02;

for(b; b>0; b--){c\*=a;}

return c;}

}

#endif /\* myName\_h \*/



// comsc 200

// main.cpp

// lab8c

// completed

// Created by Jeff on 10/5/16.

// Copyright © 2016 Jeff zhang. All rights reserved.

//

#include <iostream>

#include <algorithm>

#include <string>

#include <sstream>

#include <vector>

using namespace std;

void menu(){

cout << "\n---Rainfall statistics smart menu---\n"

<<"D display \n"

<<"N new\n"

<<"1-max\n"

<<"2-min\n"

<<"3-average\n"

<<"4-median\n"

<<"Q-to quit this program\n"

<<"enter your command : ";

}

const int NUM\_MONTH=12;

string del=","; //delimiter

int main() {

cout<<"testing all 12 monthly rainfall (in inches) delimited by comma (,) : ";

double rainFall[]= {1.10, 2, 3, 4, 5, 6, 7, 30.01, 9, 10, 11.11, 12 };

vector<double>rf(rainFall,end(rainFall));

vector<double>srf=rf;// diff srf(rf) copy contain as it were created. srf=rf default constructor ,allocated the space , make a overloading ,copy assement ,not contain, nothing ?

for(auto item:rf)

cout << item<< " ";

rf.clear();

while(true){

menu();

string choice;

cin >> choice;

cin.ignore();

switch (choice[0]) {

case 'N':

case 'n':

{ string input, token;

cout << "Enter a line of comma (,) delmited data set: ";

getline (cin, input); // user input -> string

stringstream ss(input); // string -> stream

while ( getline(ss, token, ',') )

{ // stream -> string token

stringstream sst(token); // string token -> stream token

double ff;

sst >> ff;

rf.push\_back(ff);// stream token -> double token

cout << ff << std::endl;

}

srf =rf;

sort(srf.begin(),srf.end());

break;

}

case 'D':

case 'd':

{

for(auto item:rf)

cout << item<< " , ";

rf.clear();

break;

}

case '1':

{

srf =rf;

sort(srf.begin(),srf.end());

cout<< srf.back();

break;

}

case '2':

{

srf =rf;

sort(srf.begin(),srf.end());

cout<< srf.front();

break;

}

case 's':

case 'S':

{

srf =rf;

sort(srf.begin(),srf.end());

for(auto item:srf)

cout << item << endl;

break;

}

case '3':

{

double t=0;

for(auto item:rf)t += item;

cout << t/NUM\_MONTH<<endl;

break;

}

case '4':

{

srf =rf;

sort(srf.begin(),srf.end());

size\_t n = srf.size() / 2;

std::nth\_element(srf.begin(), srf.begin()+n, srf.end());

int vn = srf[n];

if(srf.size()%2 == 1)

{

cout<< vn;

}else

{

std::nth\_element(srf.begin(), srf.begin()+n-1, srf.end());

cout<< 0.5\*(vn+srf[n-1]);

}

}break;

case 'q':

case 'Q':

{

return true;

}

default:

cout<<"wrong command ! " ;

}

}

}

