PPL LAB – 9

NAME : SS ARWINDHA SIVAM

ROLL NO. : 106120126

DATE : 28.10.2021

1. Swapping two numbers using class

#include<iostream>

#include<conio.h>

using namespace std;

class swapping

{

public:

int a,b,temp;

void swapno()

{

cout<<"Enter the numbers \n";

cin>>a>>b;

temp=a;

a=b;

b=temp;

cout<<"The numbers are "<<a<<" "<<b;

}

};

int main()

{

swapping n;

n.swapno();

return 0;

}

Graphical user interface

Description automatically generated with low confidence

1. Linear search in array

#include<iostream>

#include<conio.h>

using namespace std;

void linear\_search(int a[],int x,int n)

{

int z=0;

for(int i=0;i<n+1;i++)

{

if(x==a[i]&&i<n)

{

cout<<"The element is at "<<i+1<<"th position";

z++;

break;

}}

if(z==0)

{

cout<<"The element entered cannot be found in the array";

}

}

int main()

{

int arr[100],n,x;

cout<<"enter the number of elements in array\n";

cin>>n;

for(int i=0;i<n;i++)

{

cout<<"Enter the "<<i+1<<"th element";

cin>>arr[i];

}

cout<<"Enter the element to be found\n";

cin>>x;

linear\_search(arr,x,n);

return 0;

}

Graphical user interface

Description automatically generated with medium confidence

1. Polar Co-ordinate system

#include <iostream>

#include <math.h>

#define PI 3.14259

using namespace std;

class polar{

friend void add(polar,polar);

friend void sub(polar,polar);

friend void angle(polar,polar);

public:

float radius,angle,rad,x,y;

};

void add(polar p1, polar p2){

cout<<":Addition:"<<endl;

cout<<"X-coordinate="<<p1.x+p2.x<<endl<<"Y-coordinate="<<p1.y+p2.y<<endl;

}

void sub(polar p1, polar p2){

cout<<":Subtraction:"<<endl;

cout<<"X-coordinate="<<p1.x-p2.x<<endl<<"Y-coordinate="<<p1.y-p2.y<<endl;

}

void angle(polar p1, polar p2){

cout<<"Angle difference="<<p1.angle-p2.angle<<endl;

}

int main()

{

polar p1,p2,sum,diff;

cout<<"Enter the point1's radius and angle:";

cin>>p1.radius>>p1.angle;

cout<<"Enter the point2's radius and angle:";

cin>>p2.radius>>p2.angle;

p1.rad= p1.angle \* PI/ 180;

p2.rad= p2.angle \* PI/ 180;

p1.x=p1.radius\*(cos(p1.rad));

p2.x=p2.radius\*(cos(p2.rad));

p1.y=p1.radius\*(sin(p1.rad));

p2.y=p2.radius\*(sin(p2.rad));

add(p1,p2);

sub(p1,p2);

angle(p1,p2);

cout<<"Converting to Polar form--->";

sum.radius=abs((p1.x+p2.x)/(cos(p1.angle+p2.angle)));

diff.radius=abs((p1.y-p2.y)/(sin(p1.angle-p2.angle)));

cout<<"The Sum radius="<<sum.radius<<endl;

cout<<"The difference radius="<<diff.radius<<endl;

return 0;

}

Graphical user interface

Description automatically generated with medium confidence

1. Hospital Database

#include<iostream>

#include<conio.h>

#include<string.h>

using namespace std;

class hospital

{

private:

struct date

{

int day;

int month;

int year;

}adate,ddate;

public:

char name[100];

char disease[100];

void details()

{

cout<<"Enter the patient's name"<<endl;

cin.getline(name,100);

cout<<"Enter the disease"<<endl;

cin.getline(disease,100);

cout<<"Enter the date of admission"<<endl;

cin>>adate.day>>adate.month>>adate.year;

cout<<"Enter the date of discharge"<<endl;

cin>>ddate.day>>ddate.month>>ddate.year;

}

void display()

{

cout<<"The patient's name is: "<<name<<endl;

cout<<"The disease is: "<<disease<<endl;

cout<<"The date of admission is: "<<adate.day<<" "<<adate.month<<" "<<adate.year<<endl;

cout<<"The date of discharge is: "<<ddate.day<<" "<<ddate.month<<" "<<ddate.year<<endl;

}

};

int main()

{

hospital a,b;

a.details();

a.display();

return 0;

}

Graphical user interface, application

Description automatically generated with medium confidence

1. Rectangular Co-ordinate system

#include<iostream>

#include<conio.h>

#include<string.h>

#include<math.h>

using namespace std;

class rect

{ public:

double x1,y1,x2,y2,xs,ys,xa,ya,angle1;

void add()

{

cout<<"Enter the coordinate 1: "<<endl;

cin>>x1>>y1;

cout<<"Enter the coordinate 2: "<<endl;

cin>>x2>>y2;

xa=x1+x2;

ya=y1+y2;

cout<<"The sum of the coordinates are: "<<xa<<" "<<ya<<endl;

}

void sub()

{

cout<<"Enter the coordinate 1: "<<endl;

cin>>x1>>y1;

cout<<"Enter the coordinate 2: "<<endl;

cin>>x2>>y2;

xs=x2-x1;

ys=y2-y1;

cout<<"The difference between the two points are: "<<xs<<" "<<ys<<endl;

}

void angle()

{

angle1=atan(ys/xs);

cout<<"The angle is: "<<angle1;

}

};

int main()

{

rect a;

a.add();

a.sub();

a.angle();

return 0;

}

Application

Description automatically generated with low confidence

1. Constructor and Destructor

#include <iostream>

using namespace std;

class Rectangle {

public:

float length, breadth;

public:

Rectangle() {

cout << "\n\n\*\*\* Inside the Constructor \*\*\*\* \n\n";

length = 2;

breadth = 4;

}

public:

~Rectangle() {

cout << "\n\n\*\*\* Inside the Destructor \*\*\*\* \n\n";

}

};

int main() {

cout << "\nCalling the default Constructor of the Rectangle class to initialize the object.\n\n";

Rectangle rect;

cout << "\nThe Length of the Rectangle set by the Constructor is = " << rect.length << "\n\n";

cout << "\nThe Breadth of the Rectangle set by the Constructor is = " << rect.breadth << "\n\n";

return 0;

}

Graphical user interface, text

Description automatically generated

1. Inventory

#include<iostream>

using namespace std;

class inventory

{

private:

int prodID,qtyInStock;

string description;

public:

inventory(int prodID=100,int qtyInStock=0,string description="none")

{

this->prodID=prodID;

this->description=description;

this->qtyInStock=qtyInStock;

}

int remove\_items(int a)

{

if(qtyInStock-a<0)

return -1;

qtyInStock-=a;

return qtyInStock;

}

};

int main()

{

inventory pencils(234,45);

inventory erasers(235,50,"used to erase");

int temp1=pencils.remove\_items(40);

int temp2=erasers.remove\_items(60);

cout<<"Items remaining: ";

if(temp1>0)

cout<<temp1;

else

cout<<"Error";

cout<<endl<<"Items remaining: ";

if(temp2>0)

cout<<temp2;

else

cout<<"Error";

return 0;

}

Application

Description automatically generated with medium confidence