SHAURYA SINGH SRINET  
RA2111032010006

CSE IOT

Q1

#include <cmath>

#include <cstdio>

#include <vector>

#include <iostream>

#include <algorithm>

using namespace std;

class Person

{

protected:

int age;

string name;

public:

virtual void getdata(){};

virtual void putdata(){};

};

class Professor : public Person

{

int publication;

static int id1;

public:

void getdata()

{

cin>>name;

cin>>age;

cin>>publication;

}

void putdata()

{

cout<<name<<" "<<age<<" "<<publication<<" "<<id1<<endl;

id1++;

}

};

int Professor::id1=1;

class Student : public Person

{

int marks[6];

static int id2;

public:

int sum=0;

void getdata()

{

cin>>name;

cin>>age;

for(inti=0;i<=5;i++)

{

cin>>marks[i];

sum=sum+marks[i];

}

}

void putdata()

{

cout<<name<<" "<<age<<" "<<sum<<" "<<id2<<endl;

id2++;

}

};

int Student::id2=1;

int main()

{

int n, val;

cin>>n;

Person \*per[n];

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

{

cin>>val;

if(val == 1)

per[i] = new Professor;

}

else per[i] = new Student;

per[i]->getdata();

}

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

per[i]->putdata();

return 0;

}

Q2

#include <iostream>

using namespace std;

class base {

public:

virtual void show(){

cout<< "show base class" <<endl;

}

void print(){

cout<< "print base class" <<endl;

}

};

class derived : public base {

public:

void show(){

cout<< "show derived class" <<endl;

}

void print(){

cout<< "print derived class" <<endl;

}

};

int main(){

base\* bpointr;

derived dev;

bpointr = &dev;

bpointr->show();

bpointr->print();

}

Q3

#include <iostream>

using namespace std;

class ClassB;

class ClassA {

public:

ClassA() : numA(12) {}

private:

intnumA;

friend int add(ClassA, ClassB);

};

class ClassB {

public:

ClassB() : numB(1) {}

private:

intnumB;

friend int add(ClassA, ClassB);

};

int add(ClassAobjectA, ClassBobjectB) {

return (objectA.numA + objectB.numB);

}

int main() {

ClassAobjectA;

ClassBobjectB;

cout<< "Sum: " << add(objectA, objectB);

return 0;

}

Q4

#include <iostream>

#include <fstream>

using namespace std;

class Shape

{

public:

string name;

double width, height, radius;

public:

void set\_data (double a, double b)

{

width = a;

height = b;

}

virtual double area() = 0;

};

class Rectangle: public Shape

{

public:

double area ()

{

return (width \* height);

}

};

class Triangle: public Shape

{

public:

double area ()

{

return (width \* height)/2;

}

};

class Circle : public Shape

{

public:

double area ()

{

return 3.1415 \* (radius \* radius);

}

};

int main()

{

int N;

cin>> N;

Rectangle Rect;

Triangle Tri;

Circle Circ;

string\* S = new string[N];

if(N == 1) {

cin>>Rect.name>>Rect.height>>Rect.width;

cout<<Rect.area();

return 0;

}

else

{

for(inti = 0; i< N; i++)

{

cin>> S[i];

if(S[i] == "Rectangle")

{

cin>>Rect.height;

cin>>Rect.width;

}

else if(S[i] == "Triangle")

{

cin>>Tri.height;

cin>>Tri.width;

}

else if(S[i] == "Circle")

{

cin>>Circ.radius;

}

}

}

cout<<Rect.area() << " " <<Tri.area() << " " <<Circ.area();

delete [] S;

return 0;

}

Q5

#include <iostream>

using namespace std;

class Polygon

{

protected:

double width, height;

public:

void set\_data (double a, double b)

{

width = a;

height = b;

}

virtual double area() = 0;

};

class Rectangle: public Polygon

{

public:

double area ()

{

return (width \* height);

}

};

class Triangle: public Polygon

{

public:

double area ()

{

return (width \* height)/2;

}

};

int main ()

{

Polygon \*sPtr;

Rectangle Rect;

sPtr = &Rect;

sPtr ->set\_data (5,3);

cout<< "Area of Rectangle is " <<sPtr -> area() <<endl;

Triangle Tri;

sPtr = &Tri;

sPtr ->set\_data (4,6);

cout<< "Area of Triangle is " <<sPtr -> area() <<endl;

return 0;

}