**NAME: CHANDRIMA BHATTACHARYA**

**EXAM ROLL NO: 111308009**

**CLASS ROLL NO: HX-05**

**DEPARTMENT: I.T.**

**SEMESTER: 5**

**ASSIGNMENT 5 & 6**

ASSIGNMENT 5

1. Write a program using this pointer to find out the least number obtained among three subjects. Use ternary operator.

**SOURCE CODE:**

class leastnum

{

int a;

public:

void getdata()

{

cout<<"Enter value"<<endl;

cin>>a;

}

int senddata()

{

return a;

}

int compare(leastnum a1, leastnum a2)

{

return (this->senddata() < a1.senddata() ? (this->senddata() < a2.senddata() ? this->senddata() : a2.senddata() ) : (a1.senddata() < a2.senddata() ? a1.senddata() : a2.senddata())) ;

}

};

int main()

{

int numb;

leastnum x, y,z;

x.getdata();

y.getdata();

z.getdata();

numb= x.compare(y,z);

cout<<"Least number amongst given data= "<<numb<<endl;

}

**OUTPUT:**

Enter value

>>21

Enter value

>>4

Enter value

>>11

Least number amongst given data=4

1. Class polygon contains data member width and height and public method set\_value() to assign values to width and height. Class Rectangle and Triangle are inherited from polygon class. Both the classes contain public method calculate\_area() to calculate the area of Rectangle and Triangle. Use base class pointer to access the derived class object and show the area calculated.

**SOURCE CODE:**

#include <iostream>

#include <stdlib.h>

using namespace std;

class polygon

{

protected:

float width;

float height;

public:

void set\_value()

{

cout<<"Enter width and height"<<endl;

cin>>width>>height;

}

};

class rectangle: public polygon

{

float area;

public:

void calculate\_area()

{

area=width\*height;

cout<<"Area of rectangle= "<<area<<"unit square"<<endl;

}

};

class triangle: public polygon

{

float area;

public:

void calculate\_area()

{

area=width\*height/2;

cout<<"Area of triangle= "<<area<<"unit square"<<endl;

}

};

int main()

{

triangle t;

rectangle r;

triangle \*p;

p=&t;

p->set\_value();

p->calculate\_area();

rectangle \*s;

s=&r;

s->set\_value();

s->calculate\_area();

}

**OUTPUT:**

Enter width and height

>>4 8

Area of triangle= 16 unit square

Enter width and height

>>21 4

Area of rectangle=48 unit square

1. Write a program to create a class shape with functions to find area of and display the name of the shape and other essential component of the class. Create derived classes circle, rectangle and trapezoid each having overridden functions area and display. Write a suitable program to illustrate virtual functions.

**SOURCE CODE:**

#include <iostream>

using namespace std;

class shape

{

public:

virtual void area()

{

cout<<"I am arealess and hence massless entity!"<<endl;

}

virtual void display()

{

cout<<"I am shapeless shape!"<<endl;

}

};

class circle: public shape

{

float rad;

float a;

public:

void area()

{

cout<<"Enter radius"<<endl;

cin>>rad;

a=3.14\*rad\*rad;

}

void display()

{

cout<<"Area of circle= "<<a<<"unit square"<<endl;

}

};

class rectangle: public shape

{

float h;

float w;

float a;

public:

void area(){

cout<<"Enter height and width"<<endl;

cin>>h>>w;

a=h\*w;

}

void display(){

cout<<"Area of rectangle= "<<a<<"unit square"<<endl;

}

};

class trapezoid: public shape

{

float p1;

float p2;

float h;

float a;

public:

void area()

{

cout<<"Enter parrallel sides and height"<<endl;

cin>>p1>>p2>>h;

a=0.5\*(p1+p2)\*h;

}

void display()

{

cout<<"Area of trapezium= "<<a<<"unit square"<<endl;

}

};

int main()

{

shape \*s1, \*s2,\*s3;

circle c;

rectangle r;

trapezoid t;

s1=&c;

s2=&r;

s3=&t;

s1->area();

s1->display();

s2->area();

s2->display();

s3->area();

s3->display();

}

**OUTPUT:**

Enter radius

>>7

Area of circle= 153.86 unit square

Enter height and width

>>4 21

Area of rectangle= 48 unit square

Enter parrallel sides and height

>>2 5 8

Area of trapezium= 28 unit square

1. Write a program with Student as abstract class and create derive classes Engineering, Medicine and Science from base class Student. Create the objects of the derived classes and process them and access them using array of pointer of type base class Student.

**SOURCE CODE:**

#include <iostream>

using namespace std;

class student

{

public:

virtual void display()=0;

};

class engineering: public student{

public:

void display(){

cout<<"I'm engineering student"<<endl;

}

};

class medical: public student

{

public:

void display(){

cout<<"I'm medical student"<<endl;

}

};

class science: public student{

public:

void display(){

cout<<"I'm science student"<<endl;

}

};

int main()

{

student \*s1, \*s2, \*s3;

engineering e;

medical m;

science s;

s1=&e;

s2=&m;

s3=&s;

s1->display();

s2->display();

s3->display();

}

**OUTPUT:**

I'm engineering student

I'm medical student

I'm science student

ASSIGNMENT 6

1. Two integers are taken from keyboard. Then perform division operation. a) A try block to throw an exception when a wrong type of data is keyed. b) When division by zero occurs. write appropriate catch block to handle the exception thrown.

**SOURCE CODE:**

#include <iostream>

using namespace std;

class A

{

public:

void division(int x,int y){

cout<<"We are inside division function in class A"<<endl;

if(y>0)

cout<<(x/y)<<endl;

else

throw(y);

}

};

int main()

{

A a1, a2;

int a,b;

try{

cout<<"We are inside the try block"<<endl;

cout<<"Enter 2 numbers-"<<endl;

if(!(cin>>a>>b))

throw ("Caught!Value not and integer");

a1.division(a,b);

}

catch (int i)

{

cout<<"Caught divide by 0 exception!"<<endl;

}

catch (const char \*s)

{

cout<<s<<endl;

}

}

**OUTPUT:**

**1**.We are inside the try block

Enter 2 numbers-

>>2 S

Caught!Value not and integer

**2**.We are inside the try block

Enter 2 numbers-

>>2 0

Caught divide by 0 exception!

2. Design stack and queue classes with necessary exception handling

**SOURCE CODE:**

#include<iostream>

#define SIZE 5

using namespace std;

class Stack

{

private:

int a[SIZE];

int top;

public:

Stack()

{

top=0;

}

void push(int i)

{

try

{

if(isFull())

{

throw ("Full stack!");

}

else

{

a[top]=i;

top++;

}

}

catch(const char \*msg)

{

cout<<msg<<endl;

}

}

int pop()

{

try

{

if(isEmpty())

{

throw ("Empty stack!");

}

else

{

return(a[--top]);

}

}

catch(const char \*msg)

{

cout<<msg<<endl;

}

return 0;

}

int isEmpty(){

return(top==SIZE?0:1);

}

int isFull()

{

return (top==SIZE?1:0);

}

void display(){

if(!isEmpty())

{

for(int i=top-1;i>=0;i--)

cout<<a[i]<<endl;

}

}

};

int main()

{

Stack s;

int ch=1;

int num;

while(ch!=0)

{

cout<<"1. push"<<endl;

cout<<"2. pop"<<endl;

cout<<"3. display"<<endl;

cout<<"0. Exit"<<endl;

cout<<"Enter your choice :";

cin>>ch;

switch(ch)

{

case 0: break;

case 1: cout<<"Enter the number to push";

cin>>num;

s.push(num);

break;

case 2: cout<<"a number was popped from the stack"<<endl;

s.pop();

break;

case 3: cout<<"The numbers are"<<endl;

s.display();

break;

default:

cout<<"try again";

}

}

return 0;

}

**OUTPUT:**

1. push

2. pop

3. display

0. Exit

Enter your choice :

>>1

Enter the number to push

>>4

1. push

2. pop

3. display

0. Exit

Enter your choice:

>>3

The numbers are 4

1. push

2. pop

3. display

0. Exit

Enter your choice:

>>0

**SOURCE CODE:**

#include<iostream>

#define SIZE 5

using namespace std;

class Queue

{

private:

int rear;

int front;

int s[SIZE];

public:

Queue()

{

front=0;

rear=-1;

}

void insert(int item)

{

try

{

if(isFull())

{

throw ("Full Queue");

}

else

{

rear=rear+1;

s[rear]=item;

}

}

catch(const char \*msg)

{

cout<<msg;

}

}

void del()

{

int item;

try

{

if(isEmpty())

{

throw "Empty Queue";

}

else

{

item=s[front];

front=front+1;

cout<<"\n DELETED ELEMENT IS %d\n\n"<<item;

}

}

catch(const char \*msg)

{

cout<<msg;

}

}

int isEmpty()

{

return(front>rear?1:0);

}

int isFull()

{

return(rear==SIZE?1:0);

}

void display()

{

cout<<"\n";

for(int i=front;i<=rear;i++)

{

cout<<s[i]<<"\t";

}

}

};

int main()

{

int ch;

Queue q;

int item;

while(ch!=0)

{

cout<<"\n\n1.INSERTION \n";

cout<<"2.DELETION \n";

cout<<"0.EXIT \n";

cout<<"\nENTER YOUR CHOICE : ";

cin>>ch;

switch(ch)

{

case 1:

cout<<"\n\t INSERTION \n";

cout<<"\nENTER AN ELEMENT : ";

cin>>item;

q.insert(item);

q.display();

break;

case 2:

cout<<"\n\t DELETION \n";

q.del();

q.display();

break;

case 0:

break;

}

}

return 0;

}

**OUTPUT:**

1.INSERTION

2.DELETION

0.EXIT

ENTER YOUR CHOICE

>>0