# CPP-Assignments

**LAB – 1** :

1:write program to test Hello World.

#include<iostream>

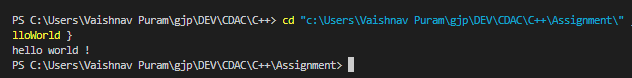
using namespace std;

int main(){

    cout<<"hello world !";

    return 0;

}



2:Write a program to adddition of two numbers .

#include<iostream>

using namespace std;

int main(){

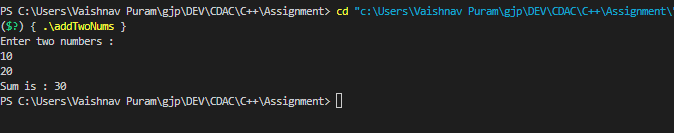
    int a ,b;

    cout<<"Enter two numbers :"<<endl;

    cin>>a>>b;

    cout<<"Sum is : "<<a+b<<endl;

}



3:Write a program to swap two numbers.

#include<iostream>

using namespace std;

int main(){

    int a,b;

    cout<<"Enter two numbers :";

    cin>>a>>b;

    cout<<"a = "<<a<<" b = "<<b<<endl;

    a=a^b;

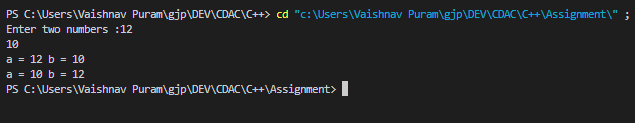
    b=a^b;

    a=a^b;

    cout<<"a = "<<a<<" b = "<<b<<endl;

    return 0;

}



4. Write a program to accept an integer and check if it is even or odd.

#include<iostream>

using namespace std;

int main(){

    int a;

    cout<<"Enter a number :"<<endl;

    cin>>a;

    if(a%2==0){

        cout<<"Even";

    }

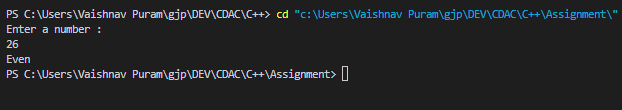
    else{

        cout<<"Odd";

    }

    return 0;

}



5. Write a program to accept a number and check if it is divisible by 5 and 7.

#include<iostream>

using namespace std;

int main(){

    int a;

    cout<<"Enter a number:"<<endl;

    cin>>a;

    if((a%5==0)&&(a%11==0)){

        cout<<"Divisible by 5 and 11";

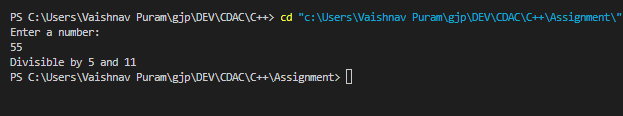
    }

    else{

        cout<<"Not divisible by 5 and 11";

    }

}



6. Write a program, which accepts annual basic salary of an employee and calculates and displays the

Income tax as per the following rules.

Basic: < 1, 50,000 Tax = 0

1, 50,000 to 3,00,000 Tax = 20%

> 3,00,000 Tax = 30%

#include<iostream>

using namespace std;

int main()

{

  float s;

  cout<<"Enter an Annual Salary : ";

  cin>>s;

  if(s>0 && s<=150000)

    {

      cout<<"No TAX";

    }

  else if(s>150000 && s<=300000)

    {

        float tax;

        tax=(s\*0.20);

        cout<<"you have to pay"<<" "<<tax<<" "<<"as a TAX";

    }

  else if( s>300000)

    {

        float tax;

        tax=(s\*0.30);

       cout<<"you have to pay"<<" "<<tax<<" "<<"as a TAX";

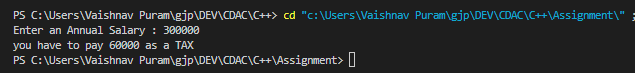
    }

  else

       cout<<"Invalid Input";

  return 0;

}



7. Accept a lowercase character from the user and check whether the character is a vowel or consonant.

(Hint: a, e, i, o, u are vowels)

#include<iostream>

#include<ctype.h>

using namespace std;

int main(){

    char c;

    cout<<"Enter a lower case charater :";

    cin>>c;

    if(c=='a'||c=='e'||c=='i'||c=='o'||c=='u'){

        cout<<"vowel";

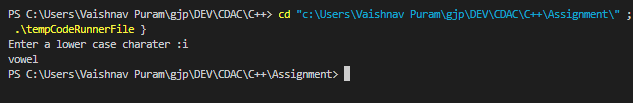
    }else{

        cout<<"consonant";

    }

    return 0;

}



8. Write a program to input angles of a triangle and check whether triangle is valid or not.

#include<iostream>

using namespace std;

int main(){

    int a,b,c;

    cout<<"Enter the three angles of the triangle : ";

    cin>>a>>b>>c;

    int sum=a+b+c;

    if(sum==180&&a!=0&&b!=0&&c!=0){

        cout<<"The given angles will form a valid triangle ";

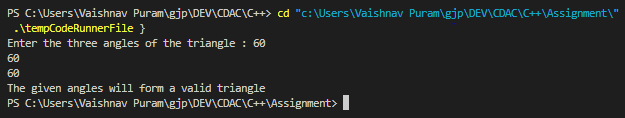
    }else{

        cout<<"The given angles will not form a valid triangle ";

    }

    return 0;

}



9:Write a program to find factorial of a given number. ex:no5 fact=5\*4\*3\*2\*1=120

#include<iostream>

using namespace std;

int main(){

    int fact=1;

    int n;

    cout<<"Enter a number:"<<endl;

    cin>>n;

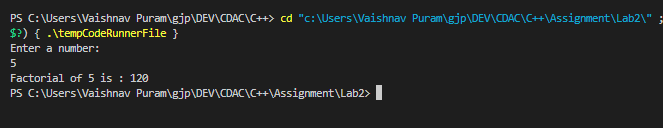
    for(int i=1;i<=n;i++){

        fact=fact\*i;

    }

    cout<<"Factorial of "<<n<<" is : "<<fact;

}



10:Write a program to find m to the power n. m=3 and n=4 so 3\*3\*3\*3

#include<iostream>

using namespace std;

int main(){

    int n,x;

    cout<<"Enter number :"<<endl;

    cin>>n;

    cout<<"Enter power :"<<endl;

    cin>>x;

    int res=1;

    for(int i=1;i<=x;i++){

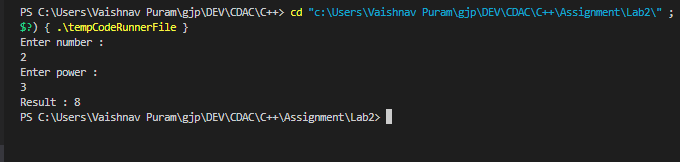
        res\*=n;

    }

    cout<<"Result : "<<res;

    return 0;

}



11:Check if number is a prime number or not.

#include<iostream>

using namespace std;

int main(){

    int n;

    cout<<"Enter number to check prime :"<<endl;

    cin>>n;

        bool flag=false;

        for(int j=2;j\*j<=n;j++){

            if(n%j==0){

                flag=true;

                break;

            }

        }

        if(!flag){

            cout<<"Prime ";

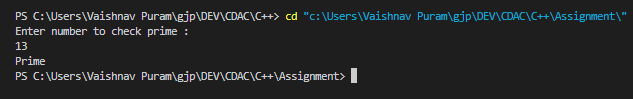
        }else{

            cout<<"Not Prime";

        }

    return 0;

}



12:Sum of series :

1+2+3+….+n

#include<iostream>

using namespace std;

int main(){

    int a;

    cout<<"Enter a number :"<<endl;

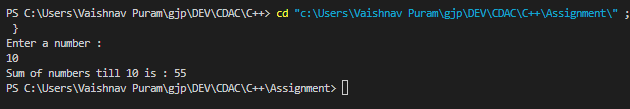
    cin>>a;

    int sum=a\*(a+1)/2;

    cout<<"Sum of numbers till "<<a<<" is : "<<sum<<endl;

    return 0;

}



13:Check whether the number is palindrome or not?

#include<iostream>

using namespace std;

int reverse(int x){

    int res=0;

    while (x>0)

    {

        int digit=x%10;

        res=res\*10+digit;

        x=x/10;

    }

    return res;

}

int main(){

    int n;

    cout<<"Enter number :"<<endl;

    cin>>n;

    int reversed=reverse(n);

    if(n==reversed){

        cout<<"Number is palindrome ";

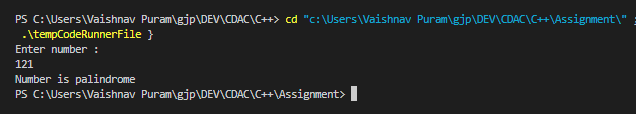
        return 0;

    }

    cout<<"Number is not a palindrome";

    return 0;

}



14:Write a program to find sum of all even and odd numbers between 1 to n.

#include<iostream>

using namespace std;

int main(){

    int eSum=0;

    int oSum=0;

    int n;

    cout<<"Enter number : "<<endl;

    cin>>n;

    for(int i=1;i<=n;i++){

        if(i%2==0){

            eSum+=i;

        }else{

            oSum+=i;

        }

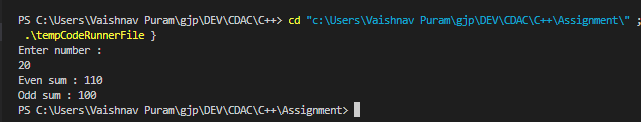
    }

    cout<<"Even sum : "<<eSum<<endl;

    cout<<"Odd sum : "<<oSum<<endl;

    return 0;

}



15: Write a program to enter a number and print its reverse.

#include<iostream>

#include<cmath>

using namespace std;

int main(){

    int n;

    cout<<"Enter a number :";

    cin>>n;

    int res=0;

    while (n>0)

    {

        int digit=n%10;

        res=res\*10+digit;

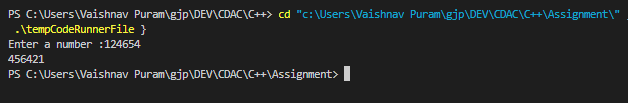
        n=n/10;

    }

    cout<<res;

    return 0;

}



16:Write a program to print all Prime numbers between 1 to n.

#include<iostream>

using namespace std;

int main(){

    int n;

    cout<<"Enter number till you want prime nos :"<<endl;

    cin>>n;

    // for any number 'n', its factors (if any) cannot be greater than its square root. So, we only need to check up to the square root of 'i'.

    for (int i = 2; i <n; i++)

    {

        bool flag=false;

        for(int j=2;j\*j<=i;j++){ //or j<=i/2 till less than equal to its square root

            if(i%j==0){

                flag=true;

                break;

            }

        }

        if(!flag){

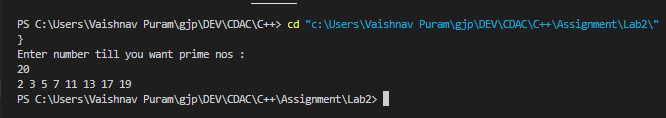
            cout<<i<<" ";

        }

    }

return 0;

}



17:Write a program to check entered number is Armstrong number or not.

#include<iostream>

#include<math.h>

using namespace std;

int amstrong(int x){

    int temp1=x;

    int temp2=x;

    int count=0;

    int res=0;

    while(temp1!=0){

        temp1=temp1/10;

        count++;

    }

    //cout<<count<<endl;

    while(temp2!=0){

        int last=temp2%10;

       // cout<<"Last : "<<last<<endl;

        int digit=1;

        for(int i=0;i<count;i++){

            digit\*=last;

        }

        //int digit=(int)pow(last,count);

       // cout<<"digit : "<<digit<<endl;

        res+=digit;

        temp2=temp2/10;

    }

    //cout<<res;

    return res;

}

int main(){

    int n;

    cout<<"Enter a number : ";

    cin>>n;

    int y=amstrong(n);

    if(n==y){

        cout<<"Amstrong number";

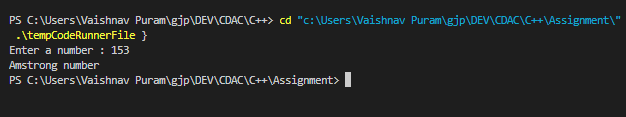
    }else{

        cout<<"Not an Amstrong number";

    }

    return 0;

}



18:Write a program to find greatest of three numbers using nested if-else.

#include<iostream>

using namespace std;

int main(){

    int a,b,c;

    cout<<"Enter three numbers :"<<endl;

    cin>>a>>b>>c;

    if(a>b){

        if(a>c){

        cout<<a<<" is the largest";

        }else{

            cout<<c<<" is the largest";

        }

    }else{

        if(b>c){

            cout<<b<<" is the largest";

        }else{

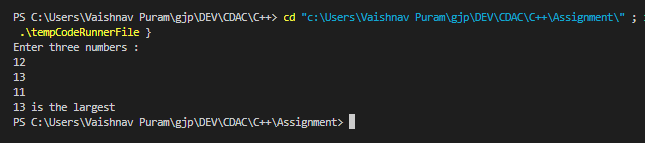
            cout<<c<<" is the largest";

        }

    }

    return 0;

}



19:Create menu driven program for Pizza Shop.And display total amount,

#include<iostream>

using namespace std;

int main()

{

    int qty=0;

    int n=0;

 int a,b,c;

 cout<<"Menu of the Pizza \n" ;

 cout<<"1.cheese Pizza=100 , 2.veggie Pizza=150 , 3.both Pizza =225 4.Exit \n"  ;

 cin>>a;

 n=a;

 cout<<"Enter the quantity of Pizza \n";

 cin>>qty;

 if(a==1)

 {

   int total;

   total=100\*qty;

   cout<<"You Ordered cheese Pizza\n";

   cout<<"Total Amount = "<<total;

 }

 else if(a==2)

 {

   int total;

   total=150\*qty;

   cout<<"You Ordered veggie Pizza\n";

   cout<<"Total Amount = "<<total;

 }

 else

 {

   int total;

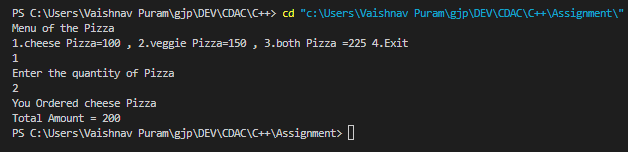
   total=250\*qty;

   cout<<"You Ordered cheese pizza and veggie Pizza \n";

   cout<<"Total Amount = "<<total;

 }

}



20:Accept a single digit from the user and display it in words. For example, if digit entered is 9, display Nine.

#include<iostream>

using namespace std;

int main(){

    int val;

    cout<<"Enter a single digit :";

    cin>>val;

    if(val>9){

        printf("Sorry you have to enter only single digit..!");

        return 0;

    }

    switch (val)

    {

    case 1:

        cout<<"One";

        break;

    case 2:

        cout<<"Two";

        break;

    case 3:

        cout<<"Three";

        break;

    case 4:

        cout<<"Four";

        break;

    case 5:

        cout<<"Five";

        break;

    case 6:

        cout<<"Six";

        break;

    case 7:

        cout<<"Seven";

        break;

    case 8:

        cout<<"Eight";

        break;

    case 9:

        cout<<"Nine";

        break;

    case 0:

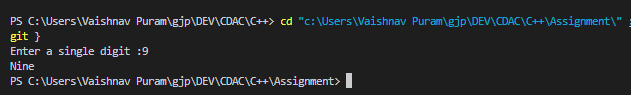
        cout<<"Zero";

        break;

    }

    return 0;

}



21. Write a program, which accepts two integers and an operator as a character (+ - \* / ), performs the corresponding operation and displays the result.

#include<iostream>

using namespace std;

int main(){

    double a,b;

    char op;

    cout<<"Enter first operand : "<<endl;

    cin>>a;

    cout<<"Enter second operand :"<<endl;

    cin>>b;

    cout<<"Enter the operator : "<<endl;

    cin>>op;

    double res;

    switch (op)

    {

    case '+':

        res=a+b;

        cout<<"Result is :"<<res<<endl;

        break;

    case '-':

        res=a-b;

        cout<<"Result is :"<<res<<endl;

        break;

    case '\*':

        res=a\*b;

        cout<<"Result is :"<<res<<endl;

        break;

    case '/':

        res=a/b;

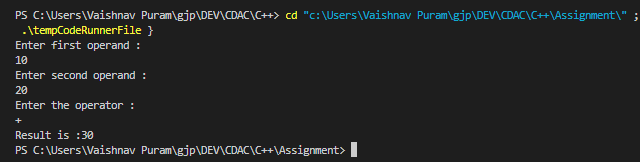
        cout<<"Result is :"<<res<<endl;

        break;

    }

    return 0;

}



**LAB – 2** :

1. Write a program that accepts numbers continuously as long as the number is positive and prints the sum of the given numbers.

#include<iostream>

using namespace std;

int main(){

    int sum=0;

    int n;

    while(n>0){

        cout<<"Enter number:";

        cin>>n;

        if(n<0){

            break;

        }

        sum+=n;

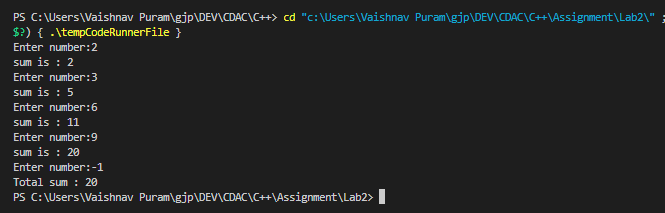
        cout<<"sum is : "<<sum<<endl;

    }

    cout<<"Total sum : "<<sum;

    return 0;

}



2. Write a program to accept two integers x and n and compute x raised to n.

#include<iostream>

using namespace std;

int main(){

    int n,x;

    cout<<"Enter number :"<<endl;

    cin>>n;

    cout<<"Enter power :"<<endl;

    cin>>x;

    int res=1;

    for(int i=1;i<=x;i++){

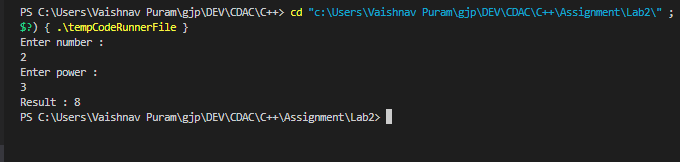
        res\*=n;

    }

    cout<<"Result : "<<res;

    return 0;

}



3. Write a program to accept a character, an integer n and display the next n characters.

#include<iostream>

using namespace std;

int main(){

    char c;

    int n;

    cout<<"Enter character :"<<endl;

    cin>>c;

    cout<<"Enter number till which you want to print characters :"<<endl;

    cin>>n;

    for(int i=1;i<=n;i++){

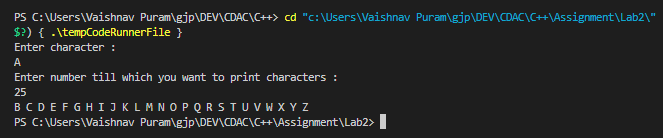
        char ch=c+i;

        cout<<ch<<" ";

    }

    return 0;

}



4. Write a program to calculate factorial of a number.

#include<iostream>

using namespace std;

int main(){

    int fact=1;

    int n;

    cout<<"Enter a number:"<<endl;

    cin>>n;

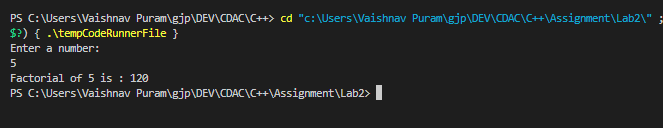
    for(int i=1;i<=n;i++){

        fact=fact\*i;

    }

    cout<<"Factorial of "<<n<<" is : "<<fact;

}



5. Write a program to calculate factors of a given number.

#include<iostream>

using namespace std;

void findFactors(int n){

    cout<<"Factors are : "<<endl;

    for(int i=1;i<=n;i++){

        if(n%i==0){

            cout<<i<<" ";

        }

    }

}

int main(){

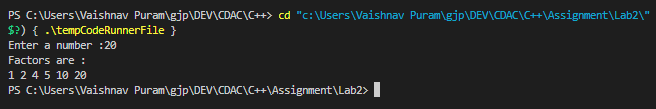
    int n;

    cout<<"Enter a number :";

    cin>>n;

    findFactors(n);

}



6. Accept two numbers and calculate GCD of them.

#include<iostream>

using namespace std;

int gcd(int a,int b){

    if(b==0){

        return a;

    }

    return gcd(b,a%b);

}

int main(){

    int a,b;

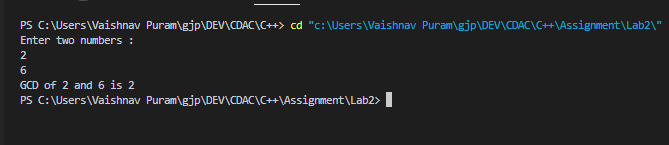
    cout<<"Enter two numbers :"<<endl;

    cin>>a>>b;

    int res=gcd(a,b);

    cout<<"GCD of "<<a<<" and "<<b <<" is " <<res<<endl;

}



7. Accept two numbers and calculate GCD of them.Write a menu driven program to do following operations :

a) Compute area of circle

b) Compute area of rectangle

c) Compute area of triangle

d) Exit

Display menu, ask choice to the user, depending on choice accept the parameters and perform the operation. Continue this process until user selects exit option.

#include<iostream>

using namespace std;

int main(){

    int n;

    do{

        cout<<"Select the options from below : "<<endl;

        cout<<"1. Area of Circle "<<endl;

        cout<<"2. Area of Rectangle"<<endl;

        cout<<"3. Area of Triangle "<<endl;

        cout<<"4. Exit "<<endl;

        cin>>n;

        switch (n)

        {

        case 1:

            /\* code \*/

            int r;

            cout<<"Enter radius of circle :";

            cin>>r;

            cout<<3.14159265\*r\*r;

            break;

        case 2:

            /\* code \*/

            int l,b;

            cout<<"Enter length and breadth of rectangle :";

            cin>>l>>b;

            cout<<l\*b;

            break;

        case 3:

            /\* code \*/

            int base,h;

            cout<<"Enter base and height of triangle";

            cin>>base>>h;

            cout<<0.5\*base\*h;

            break;

        case 4:

            /\* code \*/

            break;

        default:

            cout<<"Enter valid choice !!";

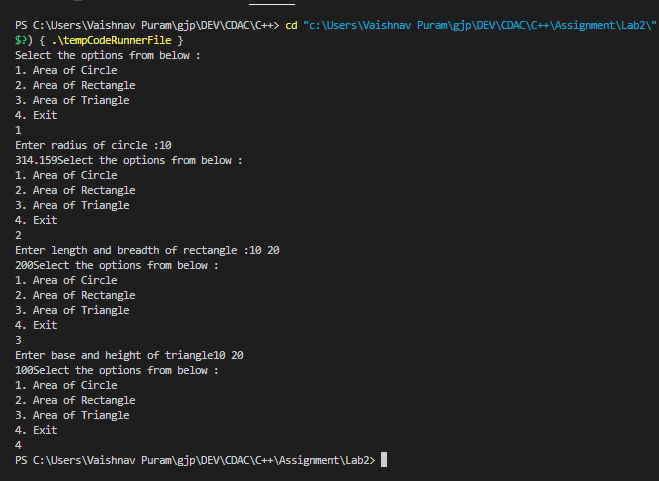
            break;

        }

    }while(n!=4);

    return 0;

}



8. Write a program to print all prime numbers between 1 to n

#include<iostream>

using namespace std;

int main(){

    int n;

    cout<<"Enter number till you want prime nos :"<<endl;

    cin>>n;

    // for any number 'n', its factors (if any) cannot be greater than its square root. So, we only need to check up to the square root of 'i'.

    for (int i = 2; i <n; i++)

    {

        bool flag=false;

        for(int j=2;j\*j<=i;j++){ //or j<=i/2 till less than equal to its square root

            if(i%j==0){

                flag=true;

                break;

            }

        }

        if(!flag){

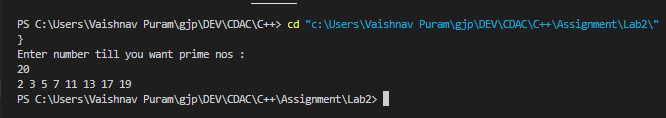
            cout<<i<<" ";

        }

    }

return 0;

}



**LAB – 3&4** :

1. Write a program to create student class with data members rollno, marks1,mark2,mark3.

Accept data (acceptInfo()) and display using display member function.

Also display total,percentage and grade.

#include<iostream>

using namespace std;

class Student{

    int rollno,marks1,marks2,marks3,total;

    double percentage;

    char grade;

    public:

        void acceptData(int rollno,int marks1,int marks2,int marks3){

            this->rollno=rollno;

            this->marks1=marks1;

            this->marks2=marks2;

            this->marks3=marks3;

        }

        char calGrade(int percentage){

            if (percentage>=75)

            {

                grade='A';

            }else if(percentage>=55&&percentage<75){

                grade='B';

            }else{

                grade='C';

            }

            return grade;

        }

        void display(){

            total=marks1+marks2+marks3;

            percentage=total/3;

            grade=calGrade(percentage);

            cout<<"Student details :"<<endl;

            cout<<"Rollno : "<<rollno<<endl;

            cout<<"Total : "<<total<<endl;

            cout<<"Percentage : "<<percentage<<endl;

            cout<<"Grade :"<<grade<<endl;

        }

};

int main(){

    int rollno,marks1,marks2,marks3;

    cout<<"Enter student details , rollno and marks :"<<endl;

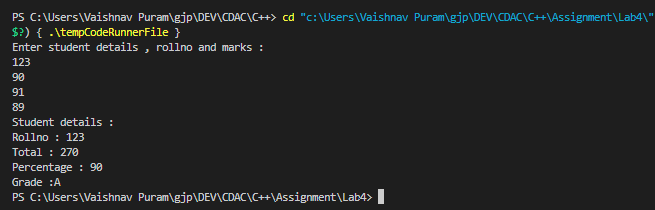
    cin>>rollno>>marks1>>marks2>>marks3;

    Student s;

    s.acceptData(rollno,marks1,marks2,marks3);

    s.display();

}



2. Create a class Person with data members as name, age, city. Write getters and setters for all the data members. Also add the display function. Create Default and Parameterized constructors. Create the object of this class in main method and invoke all the methods in that class.

#include<iostream>

using namespace std;

class Person{

    string name,city;

    int age;

    public:

        Person(){

            cout<<"-----------Default constructor-------"<<endl;

            name="Raghu";

            age=23;

            city="Bombay";

        }

        Person(string name,int age,string city){

            cout<<"---------Parameterised constructor-------------"<<endl;

            this->name=name;

            this->age=age;

            this->city=city;

        }

        void display(){

            cout<<"Name : "<<name<<endl;

            cout<<"Age : "<<age<<endl;

            cout<<"City : "<<city<<endl;

        }

        //setters

        void setName(string name){

            this->name=name;

        }

        void setAge(int age){

            this->age=age;

        }

        void setCity(string city){

            this->city=city;

        }

        //getters

        string getName(){

            return name;

        }

        int getAge(){

            return age;

        }

        string getCity(){

            return city;

        }

};

int main(){

    int age;

    string  name,city;

    cout<<"Enter person details : name , age , city :"<<endl;

    cin>>name>>age>>city;

    Person p;

    p.display();

    Person p1(name,age,city);

    p1.display();

    cout<<"Through gettters : "<<endl;

    cout<<p1.getAge()<<endl;

    cout<<p1.getName()<<endl;

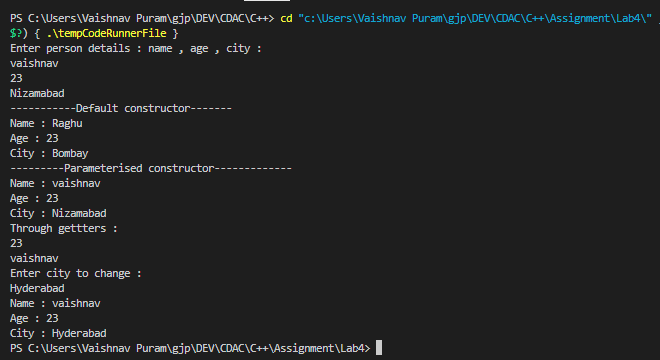
    cout<<"Enter city to change : "<<endl;

    cin>>city;

    p1.setCity(city);

    p1.display();

}



3. Create a class Date with data members as dd, mm, yy. Write getters and setters for all the data members. Also add the display function. Create Default and Parameterized constructors. Create the object of this class in main method and invoke all the methods in that class.

#include<iostream>

using namespace std;

class Date{

    int dd,mm,yy;

    public:

        Date(){

            cout<<"----------Default Constructor-----------"<<endl;

            dd=12;

            mm=03;

            yy=1947;

        }

        Date(int dd,int mm,int yy){

            this->dd=dd;

            this->mm=mm;

            this->yy=yy;

        }

        void display(){

            if(dd>31||mm>12){

                cout<<"Invalid input !!!"<<endl;

                return;

            }else{

                cout<<"Date : "<<dd<<"/"<<mm<<"/"<<yy<<endl;

            }

        }

        //setters

        void setDate(int dd){

            this->dd=dd;

        }

        void setMonth(int mm){

            this->mm=mm;

        }

        void setYear(int yy){

            this->yy=yy;

        }

        //getters

        int getDate(){

            return dd;

        }

        int getMonth(){

            return mm;

        }

        int getYear(){

            return yy;

        }

};

int main(){

    Date d;

    d.display();

    int dd,mm,yy;

    cout<<"Enter Date : "<<endl;

    cin>>dd;

    cout<<"Enter Month : "<<endl;

    cin>>mm;

    cout<<"Enter Year : "<<endl;

    cin>>yy;

    Date d1(dd,mm,yy);

    d1.display();

    char c;

    cout<<"Do you want to change the date ? (Y/N) "<<endl;

    cin>>c;

    if(c=='N'){

        return 0;

    }else{

        cout<<"Enter date,month,year :"<<endl;

        cin>>dd>>mm>>yy;

        d1.setDate(dd);

        d1.setMonth(mm);

        d1.setYear(yy);

        cout<<"Date changed ! "<<endl;

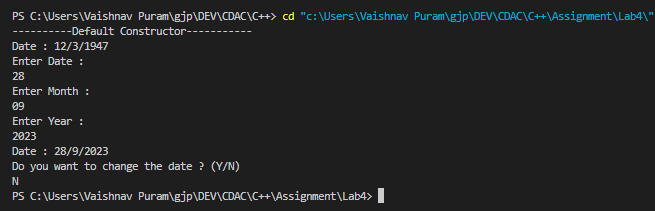
        d1.display();

        cout<<d1.getDate()<<"/"<<d1.getMonth()<<"/"<<d1.getYear();

    }

    return 0;

}



4. Create a class Book with data members as bname,id,author,price. Write getters and setters for all the data members. Also add the display function. Create Default and Parameterized constructors. Create the object of this class in main method and invoke all the methods in that class.

#include<iostream>

#include<cstring>

using namespace std;

class Book{

    string bname,author;

    int id,price;

    public:

        Book(){

        }

        Book(int id,string bname,int price,string author){

            this->id=id;

            this->bname=bname;

            this->price=price;

            this->author=author;

        }

        void setId(int id){

            this->id=id;

        }

        void setBName(string bname){

            this->bname=bname;

        }

        void setPrice(int price){

            this->price=price;

        }

        void setAuthor(string author){

            this->author=author;

        }

        string getAuthor(){

            return author;

        }

        string getBName(){

            return bname;

        }

        int getPrice(){

            return price;

        }

        int getId(){

            return id;

        }

        void display(){

            cout<<"Book Id : "<<this->getId()<<endl;

            cout<<"Book name : "<<this->getBName()<<endl;

            cout<<"Author name : "<<this->getAuthor()<<endl;

            cout<<"Price : "<<this->getPrice()<<endl;

        }

};

int main(){

    int choice;

    string author,bname;

    Book b1;

    int price;

    do{

        cout<<"----------Options-----------"<<endl;

        cout<<"1.Store a new book"<<endl;

        cout<<"2.Get the price of the book"<<endl;

        cout<<"3.Get the author of the book"<<endl;

        cout<<"4.Exit"<<endl;

        cin>>choice;

        int id=1;

        switch (choice)

        {

        case 1:

            cout<<"Enter book details : book name , author , price , :"<<endl;

            cin>>bname>>author>>price;

            b1.setBName(bname);

            b1.setAuthor(author);

            b1.setPrice(price);

            b1.setId(id++);

            b1.display();

            break;

        case 2:

            cout<<b1.getPrice()<<endl;

            break;

        case 3:

            cout<<b1.getAuthor()<<endl;

            break;

        case 4:

            break;

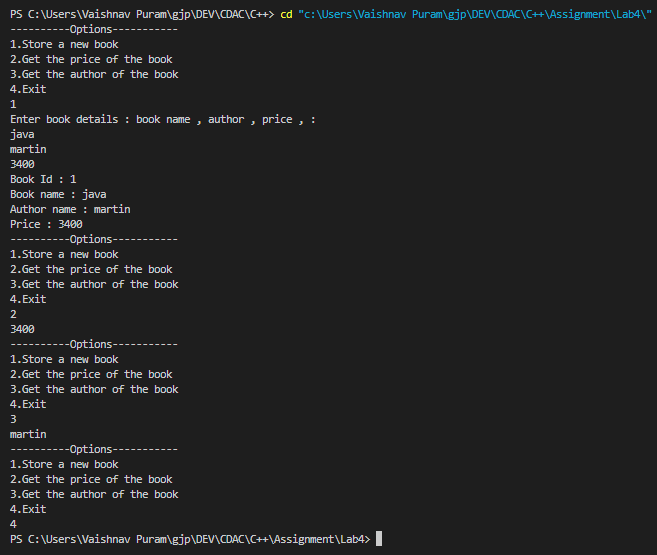
        default:

            break;

        }

    }while (choice!=4);

}



5. Create a class Point with data members as x,y. Create Default and Parameterized constructors. Write getters and setters for all the data members. Also add the display function. Create the object of this class in main method and invoke all the methods in that class.

#include<iostream>

using namespace std;

class Point{

    private:

    int x;

    int y;

    private:

    int sum(int a, int b){

        int add = a + b;

        return add;

    }

    public:

    Point(){

        cout<<"-------Default-------"<<endl;

        x=55;

        y=77;

    }

    Point(int x, int y){

        cout<<"-------Parameterised------"<<endl;

        this->x=x;

        this->y=y;

    }

    void displayDetails(){

        cout<<"x = "<<x<<", y = "<<y<<", addition is "<<sum(x,y)<<endl;

    }

    void setX(int x){

        this->x=x;

        cout<<"X has updated."<<endl;

    }

    int getX(){

        return x;

    }

    void setY(int y){

        this->y=y;

        cout<<"Y has updated."<<endl;

    }

    int getY(){

        return y;

    }

};

int main(){

    Point p1;

    p1.displayDetails();

    Point p2(5, 7);

    p2.displayDetails();

    p1.setX(9);

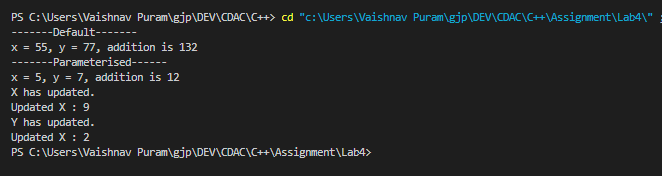
    cout<<"Updated X : "<<p1.getX()<<endl;

    p1.setY(2);

    cout<<"Updated X : "<<p1.getY()<<endl;

    return 0;

}



6. Create a class ComplexNumber with data members real, imaginary. Create Default and Parameterized constructors. Write getters and setters for all the data members. Also add the display function. Create the object of this class in main method and invoke all the methods in that class.

#include<iostream>

using namespace std;

class ComplexNumber{

    private : int real,img;

    public :

    ComplexNumber(){

        cout<<"------Default------"<<endl;

        real = 4;

        img = 9;

    }

    ComplexNumber(int real, int img){

        cout<<"------Parameterised-----"<<endl;

        this->real = real;

        this->img = img;

    }

    void displayComplexNumber(){

        cout<<"Complex Number is "<<real<<"+"<<img<<"i"<<endl;

    }

    void setReal(int real){

        this->real=real;

    }

    int getReal(){

        return real;

    }

    void setImg(int img){

        this->img=img;

    }

    int getImg(){

        return img;

    }

};

int main() {

    ComplexNumber cn1;

    cn1.displayComplexNumber();

    ComplexNumber cn2(5, 6);

    cn2.displayComplexNumber();

    cn2.setReal(2);

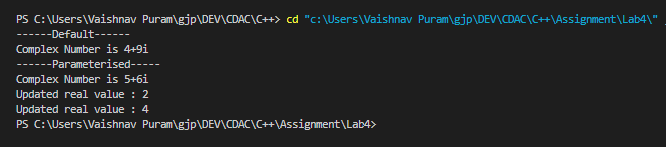
    cout<<"Updated real value : "<<cn2.getReal()<<endl;

    cn2.setImg(4);

    cout<<"Updated real value : "<<cn2.getImg()<<endl;

    return 0;

}



**LAB -5** :

1. Create Date class with members day,month ,year.

Write no argument and parameterised constructor .Create two object s and initialize them using no argument and parameterised constructor respectively. Print date using display function.

#include<iostream>

using namespace std;

class DateException{

    string msg;

    public:

        DateException(string msg){

            this->msg=msg;

        }

        void name(){

            cout<<"Exception !"<<msg<<endl;

        }

};

class Date{

    int day;

    int month;

    int year;

    public:

        Date(){

            this->day=28;

            this->month=1;

            this->year=1999;

        }

        Date(int day,int month,int year){

            this->day=day;

            this->month=month;

            this->year=year;

        }

        void display(){

            DateException e("Invalid date !");

            if(this->day>31||this->month>12){

                throw e;

            }

            cout<<day<<"/"<<month<<"/"<<year<<endl;

        }

};

int main(){

    int d,m,y;

    cout<<"Enter day,month and year : "<<endl;

    cin>>d>>m>>y;

    Date d1;

    Date d2(d,m,y);

    try{

        d1.display();

        d2.display();

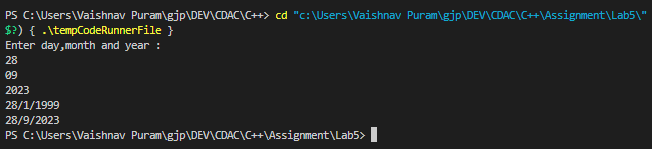
    }catch(DateException &e){

        e.name();

    }

    return 0;

}



2. Create Employee class with members id(int),name(string),dob(Date).Use above created Date class.

Write default and parameterised constructor in Employee Class.Write accept() function to accept information and display() to display emp information.

#include<iostream>

using namespace std;

class Date{

    int day;

    int month;

    int year;

    public:

        Date(){

            this->day=28;

            this->month=1;

            this->year=1999;

        }

        Date(int day,int month,int year){

            this->day=day;

            this->month=month;

            this->year=year;

        }

        void accceptDate(){

            cin>>day>>month>>year;

        }

        void display(){

            cout<<day<<"/"<<month<<"/"<<year<<endl;

        }

};

class Employee{

    int empno;

    string name;

    Date d;

    public:

        void acceptDetails(){

            cout<<"Enter emp id :\n";

            cin>>empno;

            cout<<"Enter emp nmae : \n";

            cin>>name;

            cout<<"Enter Date : \n";

            d.accceptDate();

        }

        void display(){

            cout<<"Emp Id : "<<empno<<endl;

            cout<<"Emp name : "<<name<<endl;

            d.display();

        }

};

int main(){

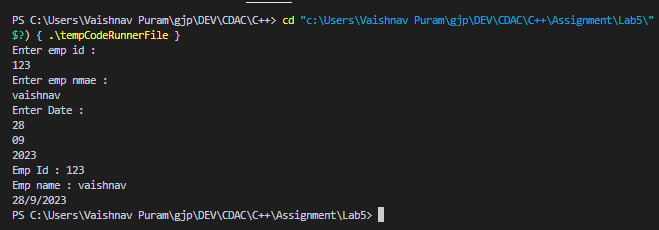
    Employee e;

    e.acceptDetails();

    e.display();

    return 0;

}



3. Consider that payroll software needs to be developed for computerization of

operations of an ABC organization. The organization has employees.

3.1. Construct a class Employee with following members using private access

specifies:

Employee Id integer

Employee Name string

Basic Salary double

HRA double

Medical double=1000

PF double

PT double

Net Salary double

Gross Salary double

Please use following expressions for calculations://Note:Don't accept HRA,PF PT from user

\* HRA = 50% of Basic Salary

\* PF = 12% of Basic Salary

\* PT = Rs. 200

3.2. Write methods to display the details of an employee and calculate the gross

and net salary.

\* Goss Salary = Basic Salary + HRA + Medical

\* Net Salary = Gross Salary – (PT + PF)

Create Object of employee class and assign values and display Details.

#include<iostream>

using namespace std;

class Employee{

    int id;

    string name;

    double sal;

    double hra;

    double medical=1000;

    double pf;

    double pt=20;

    double netSal;

    double grossSal;

    public:

        void getDetails(){

            hra=0.5\*sal;

            pf=0.12\*sal;

            grossSal=sal+hra+medical;

            netSal=grossSal-(pt+pf);

            cout<<"Employee Details : "<<endl;

            cout<<"ID : "<<id<<endl;

            cout<<"Name : "<<name<<endl;

            cout<<"Basic Salary : "<<sal<<endl;

            cout<<"Medical : "<<medical<<endl;

            cout<<"HRA : "<<hra<<endl;

            cout<<"PF : "<<pf<<endl;

            cout<<"PT : "<<pt<<endl;

            cout<<"Gross Salary : "<<grossSal<<endl;

            cout<<"Net Salary : "<<netSal<<endl;

        }

        Employee(int id,string name,double sal){

            this->id=id;

            this->name=name;

            this->sal=sal;

        }

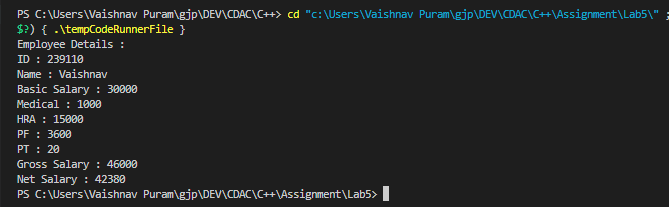
};

int main(){

    Employee e(239110,"Vaishnav",30000);

    e.getDetails();

}



**LAB -6** :

1. Create Emp based organization structure --- Emp , Mgr , Worker

1.1 Emp state--- id(int), name, deptId , basicSalary(double)

Accept all of above in constructor arguments.

Methods ---

1.2. compute net salary ---ret 0

(eg : public double computeNetSalary(){return 0;})

1.2 Mgr state ---id,name,basic,deptId , perfBonus

Add suitable constructor

Methods ----

1. compute net salary (formula: basic+perfBonus) -- override computeNetSalary

1.3 Worker state --id,name,basic,deptId,hoursWorked,hourlyRate

Methods :

1. compute net salary (formula: = basic+(hoursWorked\*hourlyRate) --override computeNetSalary

2. get hrlyRate of the worker -- add a new method to return hourly rate of a worker.(getter)

Create suitable array to store organization details.

Provide following options

1. Hire Manager

I/P : all manager details

2. Hire Worker

I/P : all worker details

3. Display information of all employees net salary (by invoking computeNetSal),

4. Exit

#include<iostream>

#include<list>

#include<typeinfo>

using namespace std;

class Employee{

    protected:

        int id;

        string name;

        int deptId;

        double basicSal;

        public:

            virtual double netSal(){

                return 0.0;

            }

            Employee(){

            }

            Employee(int id,string name,int deptId,double basicSal){

                this->id=id;

                this->name=name;

                this->deptId=deptId;

                this->basicSal=basicSal;

            }

};

class Manager:public Employee{

    protected:

    double prefBonus;

    double net;

    public:

        double netSal(){

            return net=basicSal+prefBonus;

        }

        Manager(){

        }

        Manager(int id,string name,int deptId,double basicSal,double prefBonus):Employee(id,name,deptId,basicSal){

            this->prefBonus=prefBonus;

        }

};

class Worker:public Employee{

    protected:

    int hrsWorked;

    double hrlyRate;

    double net;

    public:

        void setHrlyRate(double rate){

            hrlyRate=rate;

        }

        double getHrlyRate(){

            return hrlyRate;

        }

        double netSal(){

            return basicSal+(hrsWorked\*hrlyRate);

        }

        Worker(){

        }

        Worker(int id,string name,int deptId,double basicSal,int hrsWorked):Employee(id,name,deptId,basicSal){

            this->hrsWorked=hrsWorked;

        }

};

int main(){

    Employee\* bptr[50];

    int index=0;

    // list<Manager> manager;

    // list<Worker> worker;

    while (true) {

        cout << "Options:\n";

        cout << "1. Hire Manager\n";

        cout << "2. Hire Worker\n";

        cout << "3. Display information of all employees net salary\n";

        cout << "4. Exit\n";

        cout << "Enter your choice: ";

        int choice;

        cin >> choice;

        if(choice==4){

            break;

        }

        if(choice==1){

            int id;

            string name;

            int deptId;

            double basicSal;

            double prefBonus;

            double netSal;

            cout<<"Enter id : "<<endl;

            cin>>id;

            cout<<"Enter name : "<<endl;

            cin>>name;

            cout<<"Enter deptId : "<<endl;

            cin>>deptId;

            cout<<"Enter basic salary : "<<endl;

            cin>>basicSal;

            cout<<"Enter preferable bonus : "<<endl;

            cin>>prefBonus;

            // Manager m(id,name,deptId,basicSal,prefBonus);

            // m.netSal();

            // bptr[index++]=&m;

            Manager\* m=new Manager(id,name,deptId,basicSal,prefBonus);

            m->netSal();

            bptr[index++]=m;

        }

        if(choice==2){

            int id;

            string name;

            int deptId;

            double basicSal;

            int hrsWorked;

            double hrlyRate;

            double netSal;

            cout<<"Enter id : "<<endl;

            cin>>id;

            cout<<"Enter name : "<<endl;

            cin>>name;

            cout<<"Enter deptId : "<<endl;

            cin>>deptId;

            cout<<"Enter basic salary : "<<endl;

            cin>>basicSal;

            cout<<"Enter no.of hours worked : "<<endl;

            cin>>hrsWorked;

            cout<<"Enter hourly rate : "<<endl;

            cin>>hrlyRate;

            // Worker w(id,name,deptId,basicSal,hrsWorked);

            // w.setHrlyRate(hrlyRate);

            // w.netSal();

            // bptr[index++]=&w;

            Worker\* w=new Worker(id,name,deptId,basicSal,hrsWorked);

            w->setHrlyRate(hrlyRate);

            w->netSal();

            bptr[index++]=w;

        }

        if(choice==3){

            cout<<"Managers net salary list : "<<endl;

            for(int i=0;i<index;i++){

                if(typeid(\*bptr[i])==typeid(Manager)){

                    Manager\* m1=dynamic\_cast<Manager\*>(bptr[i]); //down casting

                double sal=m1->netSal();

                cout<<sal<<endl;

                }

            }

            cout<<"Workers net salary list : "<<endl;

            for(int i=0;i<index;i++){

                if(typeid(\*bptr[i])==typeid(Worker)){

                    Worker\* w1=dynamic\_cast<Worker\*>(bptr[i]); //down casting

                double sal=w1->netSal();

                cout<<sal<<endl;

                }

            }

        }

    }

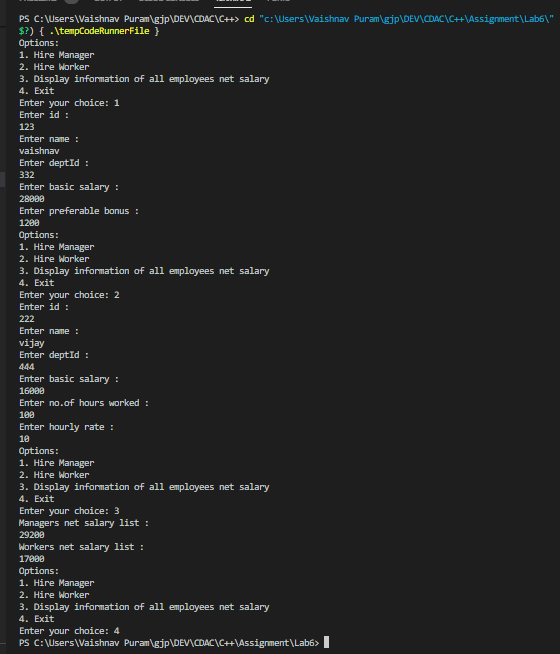
    for (int i = 0; i < index; i++) {

        delete bptr[i];

    }

    return 0;

}



2. Create cpp application for bank account handling.

2.1. Create a class BankAccount -- acct no(int),customer name(string),balance(double)

Add constr. (2 constrs : first to accept all details )

2.2 Add Business logic methods

Methods

public void withdraw(double amt)

public void deposit(double amt)

2.3: Create object of account class and test withdraw and deposit methods.

#include<iostream>

using namespace std;

class Bank{

    int acc;

    string name;

    double bal;

    public:

    Bank(){

    }

    Bank(int acc,string name,double bal){

        this->acc=acc;

        this->name=name;

        this->bal=bal;

    }

    void withdrawal(int amount){

        if(amount>0&&amount<bal){

            bal=bal-amount;

            cout<<"Withdrawed amount is : "<<amount<<endl;

            cout<<"Current balance is : "<<bal<<endl;

        }else if(amount<0||amount>bal){

            cout<<"Insufficient balance !!! "<<endl;

        }

        return;

    }

    void deposit(int amount){

        if(amount>0){

            bal=bal+amount;

            cout<<"Successfully amount : "<<amount<<" credited ! "<<endl;

            cout<<"Current balance is : "<<bal<<endl;

        }else if(amount<0){

            cout<<"Please enter amount correctly ! "<<endl;

        }

        return;

    }

    int getBalance(){

        return bal;

    }

    void setAccno(int accno){

        this->acc=accno;

    }

    void setName(string name){

        this->name=name;

    }

    void setBal(double bal){

        this->bal=bal;

    }

};

int main(){

    int accno,choice,amount;

    string name;

    double bal;

    Bank b;

    // cout<<"Enter details to create a/c : "<<endl;

    // Bank b(123,"Vaishnav",30000);

    // b.deposit(400);

    // b.getBalance();

    // b.withdrawal(400);

    do{

        cout<<"------------Options-----------"<<endl;

        cout<<"1.Create a/c "<<endl;

        cout<<"2.Withdraw "<<endl;

        cout<<"3.Deposit "<<endl;

        cout<<"4.Exit "<<endl;

        cin>>choice;

        switch (choice)

        {

        case 1:

            /\* code \*/

            cout<<"Enter a/c no : "<<endl;

            cin>>accno;

            cout<<"Enter Name : "<<endl;

            cin>>name;

            cout<<"Enter Balance : "<<endl;

            cin>>bal;

            // Bank b(accno,name,bal);

            b.setAccno(accno);

            b.setName(name);

            b.setBal(bal);

            cout<<"A/C created successfully ! "<<endl;

            break;

        case 2:

            cout<<"Enter the amount to withdraw : "<<endl;

            cin>>amount;

            b.withdrawal(amount);

            break;

        case 3:

            cout<<"Enter the amount to deposit : "<<endl;

            cin>>amount;

            b.deposit(amount);

            //depositHelper(amount);

            break;

        case 4:

            break;

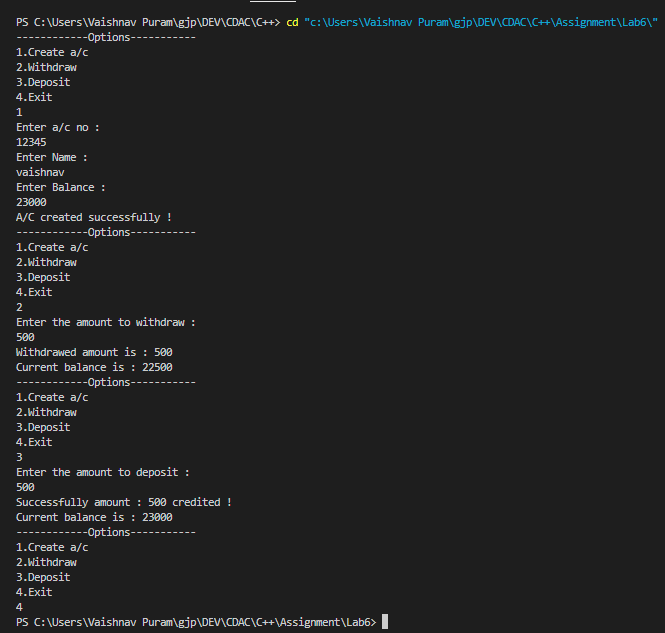
        default:

            break;

        }

    }while (choice!=4);

}



3. Create a abstract class Shape with pure virtual method area;

Create Rectangle,Circle,Square class..inherit them from Shape class..Override area method.

Test these all classes by creating object of respective class.

#include<iostream>

using namespace std;

class Shape{

    public:

        virtual void area()=0;

};

class Reactangle:public Shape{

    int l,b;

    public:

        Reactangle(int l,int b){

            this->l=l;

            this->b=b;

        }

        void area(){

            cout<<"Area of rectangle : "<<l\*b<<endl;

        }

};

class Circle:public Shape{

    int r;

    public:

        Circle(int r){

            this->r=r;

        }

        void area(){

            cout<<"Area of circle : "<<3.14\*r\*r<<endl;

        }

};

class Square:public Shape{

    int s;

    public:

        Square(int s){

            this->s=s;

        }

        void area(){

            cout<<"Area of square : "<<s\*s<<endl;

        }

};

int main(){

    //Shape s; object creation is not allowed

    Circle c(10);

    c.area();

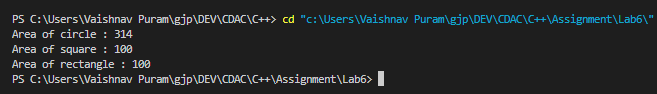
    Square s(10);

    s.area();

    Reactangle r(10,10);

    r.area();

}



**LAB – 7 :**

1:Create class for Handling Exception for Student

Create StudentException class;

Create Student class with data members (rollno,name,age) at the time of student creation check age if age<18 then throw exception

#include<iostream>

using namespace std;

class StudentException{

    string msg;

    public:

        StudentException(string msg){

            this->msg=msg;

        }

        void name(){

            cout<<msg<<endl;

        }

};

class Student{

    int rollno;

    string name;

    int age;

    public:

        Student(int rollno,string name,int age){

            this->rollno=rollno;

            this->name=name;

            this->age=age;

        }

        void display(){

            if(age<18){

                StudentException ex("Invalid age !");

                throw ex;

            }else{

                cout<<"Rollno : "<<rollno<<" Name : "<<name<<" "<<" Age : "<<age<<endl;;

            }

        }

};

int main(){

    int rollno,age;

    string name;

    cout<<"Enter student roll no :"<<endl;

    cin>>rollno;

    cout<<"Enter student name :"<<endl;

    cin>>name;

    cout<<"Enter student age : "<<endl;

    cin>>age;

    Student s(rollno,name,age);

    try{

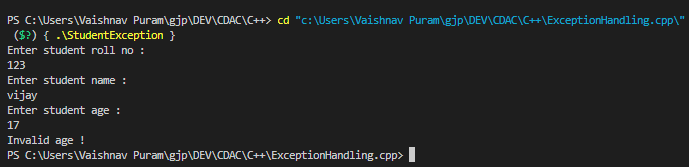
        s.display();

    }catch(StudentException &e){

        e.name();

    }

}



2:Create Function Template and class template and use it.

#include<iostream>

using namespace std;

template<class T>

void Swap(T &a,T &b){

    T temp=a;

    a=b;

    b=temp;

}

int main(){

    int a=5,b=7;

    cout<<a<<" - "<<b<<endl;

    Swap(a,b);

    cout<<a<<" - "<<b<<endl;

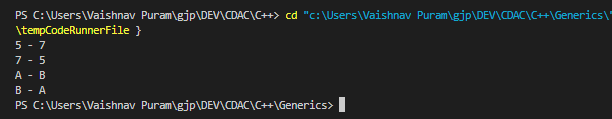
    char c='A',d='B';

    cout<<c<<" - "<<d<<endl;

    Swap(c,d);

    cout<<c<<" - "<<d<<endl;

}



#include <iostream>

using namespace std;

template<class T>

class A

{

    public:

    T num1 = 5;

    T num2 = 6;

    void add()

    {

        cout << "Addition of num1 and num2 : " << num1+num2<<endl;

    }

};

int main()

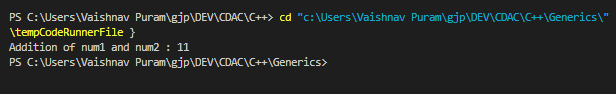
{

    A<int> d;

    d.add();

    return 0;

}



3:Create abstract class Fruit with abstract function taste

Create derived classes for Fruit:Apple,Mango,Orange

Implement abstract method in derived class ,add specific methods like juice(),jam(),pulp() in respective derived class(as discussed in lect);

create tester program.create fruit basket and invoke method

Use:dynamic\_cast,typeid

#include<iostream>

#include<typeinfo>

using namespace std;

class Fruit{

    public: string name;

            string color;

            float weight;

            Fruit(string name,string color,float weight){

            }

    public: virtual void taste()=0;

};

class Mango:public Fruit{

    public: Mango(string name,string color,float weight):Fruit(name,color,weight){

                    }

            void taste(){

                cout<<"Mango has sweet taste "<<endl;

            }

            void pulp(){

                cout<<"making pulp..."<<endl;

            }

};

class Apple:public Fruit{

    public: Apple(string name,string color,float weight):Fruit(name,color,weight){

                    }

            void taste(){

                cout<<"Apple has sweet taste "<<endl;

            }

            void jam(){

                cout<<"making jam..."<<endl;

            }

};

class Orange:public Fruit{

    public: Orange(string name,string color,float weight):Fruit(name,color,weight){

                    }

            void taste(){

                cout<<"Orange has sweet taste "<<endl;

            }

            void juice(){

                cout<<"making juice..."<<endl;

            }

};

int main(){

    Mango m("Mango","Yellow",1);

    Apple a("Apple","Red",1.75);

    Orange o("Orange","Orange",2);

    Fruit \*p[3];

    p[0]=&m;

    p[1]=&a;

    p[2]=&o;

    for(int i=0;i<3;i++){

        p[i]->taste();

        if( typeid(\*p[i]) == typeid(Mango) )

           dynamic\_cast<Mango\*>(p[i])->pulp();

        else if( typeid(\*p[i]) == typeid(Apple) )

           dynamic\_cast<Apple\*>(p[i])->jam();

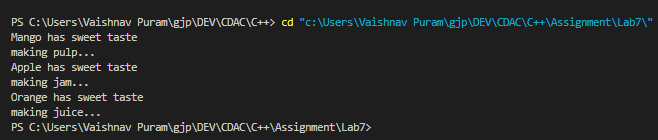
        else if(typeid(\*p[i])==typeid(Orange) )

            dynamic\_cast<Orange\*>(p[i])->juice();

    }

    return 0;

}



**LAB -8** :

1. Write a program to store characters 'A' to 'Z' in the file . Display the contents of file.

#include<iostream>

#include<fstream>

using namespace std;

int main(){

    char start,end;

    cout<<"Enter first character : "<<endl;

    cin>>start;

    cout<<"Enter last character : "<<endl;

    cin>>end;

    fstream inputfile("char2.txt",ios::out);

        for (char i = start; i <=end; i++)

        {

            /\* code \*/

            inputfile<<i<<" ";

        }

    inputfile.close();

    return 0;

}

2. Create class Employee with data members as empno,name and salary.

Accept values from user. Store it in file.

Display the contents from file.Program should be able to store information of multiple employees.

#include <iostream>

#include <fstream>

#include <cstring>

using namespace std;

class Employee {

    int empno;

    char name[100];

    double sal;

public:

    Employee() {}

    // Employee(int empno, const char\* name, double sal) {

    //     this->empno = empno;

    //     strncpy(this->name, name, sizeof(this->name));

    //     this->name[sizeof(this->name) - 1] = '\0';

    //     this->sal = sal;

    // }

    void setEmpno(int empno) {

        this->empno = empno;

    }

    void setName(const char\* name) {

        strncpy(this->name, name, sizeof(this->name));

        this->name[sizeof(this->name) - 1] = '\0';

    }

    void setSal(double sal) {

        this->sal = sal;

    }

    int getEmpno() {

        return empno;

    }

    const char\* getName() {

        return name;

    }

    double getSal() {

        return sal;

    }

    void display() {

        cout << "---------------------------" << endl;

        cout << "Emp no : " << empno << endl;

        cout << "Emp name : " << name << endl;

        cout << "Emp sal : " << sal << endl;

    }

};

int main() {

    int choice;

    Employee e;

    ofstream outputfile;

    ifstream inputfile;

    int empno;

    double salary;

    char name[100];

    do {

        cout << "--------------Options-------------" << endl;

        cout << "1.Create employee" << endl;

        cout << "2.Display Employees" << endl;

        cout << "3.Exit" << endl;

        cin >> choice;

        switch (choice) {

            case 1:

                outputfile.open("employee1.dat", ios::app | ios::binary);

                cout << "Enter emp no. : " << endl;

                cin >> empno;

                cout << "Enter emp name : " << endl;

                cin.ignore(); // Clear newline character from previous input

                cin.getline(name, sizeof(name)); // Read name as a line

                cout << "Enter emp sal : " << endl;

                cin >> salary;

                e.setEmpno(empno);

                e.setName(name);

                e.setSal(salary);

                if (!outputfile) {

                    cout << "Error in opening file!" << endl;

                    break;

                }

                outputfile.write((char\*)&e, sizeof(e));

                cout << "Employee details stored!" << endl;

                outputfile.close();

                break;

            case 2:

                inputfile.open("employee1.dat", ios::in | ios::binary);

                if (!inputfile.is\_open()) {

                    cout << "Error! No data found!!!" << endl;

                    break;

                }

                cout << "Employee details: " << endl;

                while (inputfile.read((char\*)&e, sizeof(e)) && !inputfile.eof()) {

                    e.display();

                }

                inputfile.close();

                break;

            case 3:

                break;

            default:

                break;

        }

    } while (choice != 3);

    return 0;

}



3. Write a program to copy the contents of one file and write it into another file and print it.

#include<iostream>

#include<fstream>

#include<string>

using namespace std;

int main(){

    fstream inputfile("char2.txt",ios::in);

    fstream outputfile("char3.txt",ios::out);

    string line;

    while (getline(inputfile,line))

    {

        /\* code \*/

        outputfile<<line;

    }

    inputfile.close();

    outputfile.close();

    return 0;

}

4. Create a C++ program that does the following:

Initializes an empty vector of integers.

Asks the user to enter a series of integers and adds them to the vector until the user enters a specific sentinel value (e.g., -1).

Prints the elements of the vector.

Calculates and prints the sum and average of the values in the vector.

#include<iostream>

#include<vector>

using namespace std;

int main(){

    int n;

    int sum=0;

    float avg,total;

    vector<int>v;

    do{

        cout<<"----------options----------"<<endl;

        cout<<"1.Enter number into vector : "<<endl;

        cout<<"2.Print Elements : "<<endl;

        cout<<"3.Sum of the vector elements : "<<endl;

        cout<<"4.Average of the vector elements:"<<endl;

        cout<<"-1.Exit"<<endl;

        cin>>n;

        switch(n){

            case 1:

                int num;

                cout<<"Enter number to insert into vector "<<endl;

                cin>>num;

                v.push\_back(num);

                break;

            case 2:

                // for (auto it=v.begin();it!=v.end();it++)

                // {

                //     /\* code \*/

                //     cout<<\*(it)<<" ";

                // }

                for(auto it:v){

                    cout<<it<<" ";

                }

                cout<<"\n";

                break;

            case 3:

                for (auto it:v)

                {

                    sum+=it;

                }

                cout<<"Sum of elements is : "<<sum<<endl;

                sum=0;

                break;

            case 4:

                total=v.capacity();

                for (auto it:v)

                {

                    sum+=it;

                }

                avg=sum/total;

                cout<<"Average of elemetns is : "<<avg<<endl;

                sum=0;

                break;

            default:

                break;

        }

    }while(n!=-1);

}



5. Define a Student class with attributes like name, age, and grade.

Initialize a vector of Student objects.

Allow the user to add new students to the vector with their name, age, and grade.

Print the list of students.

#include<iostream>

#include<vector>

using namespace std;

class Student{

    string name;

    int age;

    char grade;

    public:

        void setName(string name){

            this->name=name;

        }

        void setAge(int age){

            this->age=age;

        }

        void setGrade(char grade){

            this->grade=grade;

        }

        int getAge(){

            return this->age;

        }

        string getName(){

            return this->name;

        }

        char getGrade(){

            return this->grade;

        }

};

int main(){

    int n;

    int age;

    string name;

    char grade;

    Student s;

    vector<Student>v;

    do{

        cout<<"1.Enter student details "<<endl;

        cout<<"2.Display details "<<endl;

        cout<<"3.Exit"<<endl;

        cin>>n;

        switch (n)

        {

        case 1:

            cout<<"Enter name of the student"<<endl;

            cin>>name;

            cout<<"Enter age "<<endl;

            cin>>age;

            cout<<"Enter grade "<<endl;

            cin>>grade;

            s.setAge(age);

            s.setGrade(grade);

            s.setName(name);

            v.push\_back(s);

            /\* code \*/

            break;

        case 2:

            for (auto i :v)

            {

                /\* code \*/

                cout<<i.getName()<<endl;

                cout<<i.getAge()<<endl;

                cout<<i.getGrade()<<endl;

                cout<<"----------------------\n";

            }

            break;

        case 3:

            break;

        default:

            break;

        }

    }while(n!=3);

}

