**ASSIGMENT-1**

**Q-wap to show call by value ?**

**Ans-**#include <iostream>

using namespace std;

void vswap(int c, int d) ;

int main()

{

int a,b;

cout<<"call by value\n";

cout << "enter two digits" << '\n';

cin>>a;

cin>>b;

vswap(a,b);

cout << "outside function" <<"\t"<<"a="<<a<<"\t"<<"b="<<b<<"\n";

return 0;

}

void vswap(int c, int d)

{

int temp;

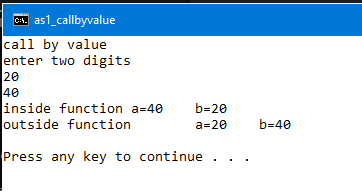
temp=c;

c=d;

d=temp;

cout<<"inside function"<<"\t"<<"a="<<c<<"\t"<<"b="<<d<<"\n";

}



**Q-wap to show call by refrence**

**Ans-**#include <iostream>

using namespace std;

void rswap(int &c, int &d) ;

int main()

{

int a,b;

cout<<"call by address\n";

cout << "enter two digits" << '\n';

cin>>a;

cin>>b;

rswap(a,b) ;

cout << "outside function" << "\t" <<"a="<<a<<"\t"<<"b="<<b<<"\n";

return 0;

}

void rswap(int &c, int &d)

{

int temp;

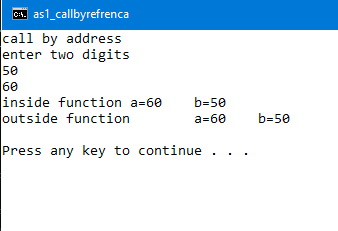
temp=c;

c=d;

d=temp;

cout<<"inside function" << "\t" <<"a="<< c <<"\t"<<"b="<< d <<"\n";

}



**Q- wap to show call by address**

#include <iostream>

using namespace std;

void aswap(int \*c, int \*d) ;

int main()

{

int a,b;

cout<<"call by address\n";

cout << "enter two digits" << '\n';

cin>>a;

cin>>b;

aswap(&a,&b) ;

cout << "outside function" << "\t" <<"a="<<a<<"\t"<<"b="<<b<<"\n";

return 0;

}

void aswap(int \*c, int \*d)

{

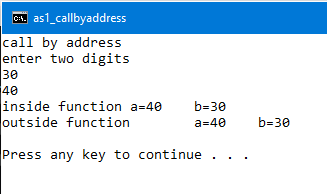
int temp;

temp=\*c;

\*c=\*d;

\*d=temp;

cout<<"inside function" << "\t" <<"a="<< \*c <<"\t"<<"b="<< \*d <<"\n";}



**Q-wap to do bank operation calculate rate of interest on fixed amount balance and fixed account balance?**

Ans-

#include<iostream>

using namespace std;

#define sar 4.3

#define fdr 6.9

void calculatesar();

void calculatefdr();

int main()

{

int c;

cout<<"WELCOME SIR \n";

cout<<"press 1). to know your saving account balance after current financial year\n";

cout<<"press 2). to know your fixed deposit balance after current financial year\n";

cout<<"please enter your choice \n";

cin>>c;

switch (c) {

case 1:

cout<<"your saving acount balance before financial year is = 2,00,000\n";

calculatesar();

break;

case 2:

cout<<"your fixed deposit amount before financial year is = 3,00,000\n";

calculatefdr();

break;

}

return 0 ;

}

void calculatesar()

{

float si ;

float amount;

float sab=200000 ;

si = ( sab\*sar\*1)/100;

amount=sab+si;

cout<<"after calculating intrest \n";

cout << "your intrest is =\t"<<si<<'\n';

cout << "your saving account balance is=\t" <<amount<< '\n';

}

void calculatefdr()

{

float si ;

float amount;

float fdb=300000 ;

si = ( fdb\*sar\*1)/100;

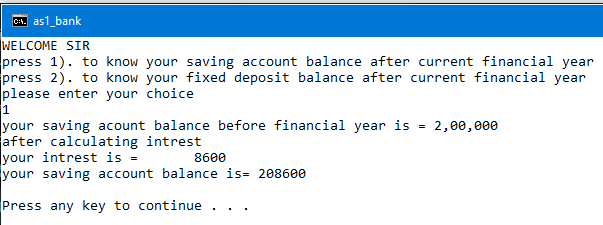
amount=fdb+si;

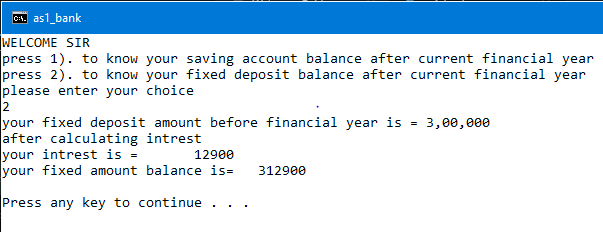
cout<<"after calculating intrest \n";

cout << "your intrest is =\t"<<si<<'\n';

cout << "your fixed amount balance is=\t" <<amount<< '\n';

}





**ASSIGMENT-2**

**Q-wap to show buble short?**

**Ans-**

#include<iostream>

using namespace std;

void insertion(int arr[],int n);

void sort(int arr[],int n);

void output(int arr[],int n);

int main()

{

int array[100], n;

cout<<"Enter number of elements\n";

cin>>n;

insertion(array,n);

sort(array,n);

output(array,n);

return 0;

}

void insertion(int arr[],int n)

{

int i;

cout<<"enter the elements"<<'\n'<< endl ;

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

{

cout<<"enter the :"<<i+1<<"element :";

cin>>arr[i];

}

}

void sort(int arr[],int n)

{

int i,j,temp;

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

{

for (j = 0 ; j < n - i - 1; j++)

{

if (arr[j] > arr[j+1]) /\* For decreasing order use < \*/

{

temp = arr[j];

arr[j] = arr[j+1];

arr[j+1] = temp;

}

}

}

}

void output(int arr[],int n)

{

int c;

cout<<"Sorted list in ascending order:\n";

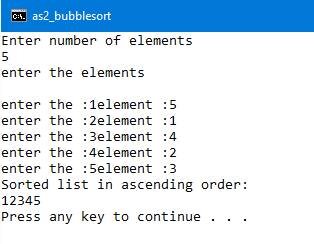
for ( c = 0 ; c < n ; c++ )

{

cout<<arr[c];

}

}



**Q-wap to implement insertion sort**

Ans-

#include<iostream>

using namespace std;

void insertion(int arr[],int n)

{

int i;

cout<<"Enter %d integers\n"<<'\n';

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

{

std::cout << "enter the \t"<<i+1<<" integer:";

cin>>arr[i];

}

}

void isort(int arr[],int n)

{

int i,j,temp;

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

{

j = i;

while ( j > 0 && arr[j-1] > arr[j])

{

temp = arr[j];

arr[j] = arr[j-1];

arr[j-1] = temp;

j--;

}

}

}

void output(int arr[],int n)

{

int c;

cout<<"Sorted list in ascending order:\n";

for (c = 0; c <n; c++)

{

cout<<arr[c];

}

}

int main()

{

int n, array[1000] ;

cout<<"Enter number of elements\n";

cin>>n;

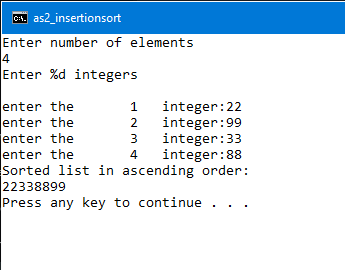
insertion(array,n);

isort(array,n);

output(array,n);

return 0;

}



**Q-wap o convert a string into upper case by taking array globally?**

Ans-

#include<iostream>

#include<string.h>

#include <stdio.h>

using namespace std;

char array[50];

void insert();

void upper();

void show();

int main()

{

cout << "welcome to the program \n to convert a string into upper case" << '\n';

insert();

upper();

show();

}

void insert()

{

char ch ;

int i=0;

cout << "enter the string you want to convert" << '\n';

do {

ch=getchar();//or we can use gets() but thats dangereous

array[i]=ch;

i++;

} while(ch!='\n');

array[i+1]='\0';

}

void upper()

{

int i;

std::cout << "before conversion" << '\n';

puts(array);

for(i=0;array[i]!='\0';i++)

{

array[i]=toupper(array[i]);

}

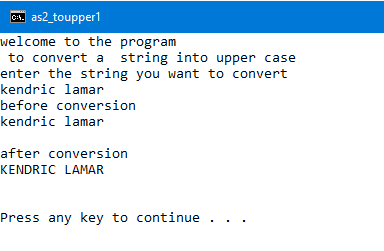
}

void show()

{

cout << "after conversion" << '\n';

puts(array);}



**Q-wap to convert a string into upper case by taking array locally?**

**Ans-**

#include<iostream>

#include<string.h>

#include <stdio.h>

using namespace std;

void insert(char array[]);

void upper(char array[]);

void show(char array[]);

int main()

{

char arr[50];

cout << "welcome to the program \n to convert a string into upper case" << '\n';

insert(arr);

upper(arr);

show(arr);

return 0;

}

void insert(char array[])

{

char ch ;

int i=0;

cout << "enter the string you want to convert" << '\n';

do {

ch=getchar();//or we can use gets() but thats dangereous

array[i]=ch;

i++;

} while(ch!='\n');

array[i+1]='\0';

}

void upper(char array[])

{

int i;

for(i=0;array[i]!='\0';i++)

{

array[i]=toupper(array[i]);

}

}

void show(char array[])

{

int i;

cout<<"after conversion\n";

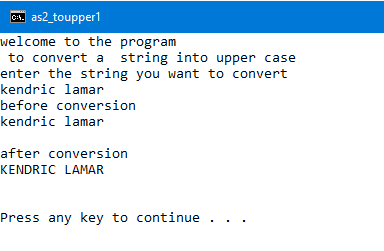
for(i=0;i<strlen(array);i++)

{

cout << array[i];

}

}



**Q-wap to convert a string into upper case by accessing array through pointer?**

**Ans-**

#include<iostream>

#include<string.h>

#include <stdio.h>

using namespace std;

void upper(char \*a);

int main()

{

char arr[50];

cout << "welcome to tthe program \n to convert a string into upper case" << '\n';

cout<<"enter the string you want to convert\n";

cin.getline(arr,50,'\n');

upper(arr);

return 0;

}

void upper(char \*a)

{

int i;

for(i=0;\*(a+i)!='\0';i++)

{

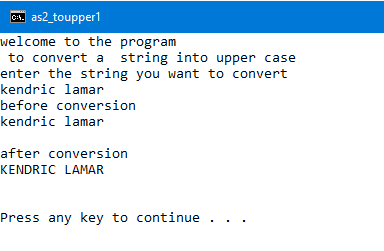
\*(a+i)=toupper(\*(a+i));

}

cout << "after conversion of string" << '\n';

for(i=0;\*(a+i);i++)

{ cout << \*(a+i);}}



**Q-wap to convert a string into upper case by returning array from function?**

**Ans-**

#include<iostream>

#include<string.h>

#include <stdio.h>

using namespace std;

char \*upper(char \*a);

int main()

{

char arr[50];

int i;

cout << "welcome to tthe program \n to convert a string into upper case" << '\n';

cout<<"enter the string you want to convert\n";

cin.getline(arr,50,'\n');

char \*ch=upper(arr);

cout << "after the conversion of array" << '\n';

for(i=0;i<strlen(arr);i++)

{

cout<<\*(ch+i);

}

return 0;

}

char \*upper(char \*a)

{

static char output[50];

int i;

for(i=0;\*(a+i)!='\0';i++)

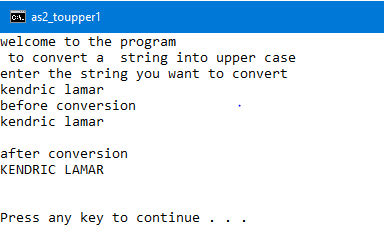
{

output[i]=toupper(\*(a+i));

}

return output;

}



**ASSIGMENT-3**

**Q-wap to create a point class and show the functioning of different constructors, access modifiers, copy constructor**

**ANS-**

#include <iostream>

using namespace std;

class tpoint

{

private:

int x;

int y;

public:

int getx();//t1 getx

int gety();//t1 gety

void print();

void modify(int a,int b);//t1 modify

tpoint();//t1 user defined defualt constructor with 0 parameters

tpoint(int a,int b);//c1 user defined defualt constructor with 2 parameters

tpoint(const tpoint &c)//c2 user defined copy constructor with copy value of c0;

{

cout << "inside copy constructor" << '\n';

x=c.x;

y=c.y;

cout << "value of x: "<< x << '\n';

cout << "value of y: "<< y << '\n';

}

};

int tpoint::getx()

{

cout << "enter the value of x" << '\n';

cin>>x;

return x;

}

int tpoint::gety()

{

cout << "enter the value of y" << '\n';

cin>>y;

return y;

}

void tpoint::print()

{

cout << "printing of values" << '\n';

cout << "x: "<< x << '\n';

cout << "y: "<< y << '\n';

}

void tpoint::modify(int a, int b)

{

cout << "modifying of the cordinates\n";

x+=a;

y+=b;

}

tpoint::tpoint()

{

cout << "inside user defined defualt constructor with 0 parameters" << '\n';

x=0;

y=0;

print();

}

tpoint::tpoint(int a,int b)

{

cout << "inside user defined defualt constructor with 2 parameters" << '\n';

x=a;

y=b;

print();

}

int main()

{

int p,q,a,b;

tpoint t1;//calling of default constructor

tpoint c1(20,40);//call to user defined defualt constructor with 2 parameters

p=t1.getx();

cout <<"value of x set to x: "<< p << '\n';

q=t1.gety();

cout <<"value of y set to y:"<< q << '\n';

cout<<"modifying value of cordinates\n";

cout << "increase the value of x by" << '\n';

cin>>a;

cout << "increase the value of y by" << '\n';

cin>>b;

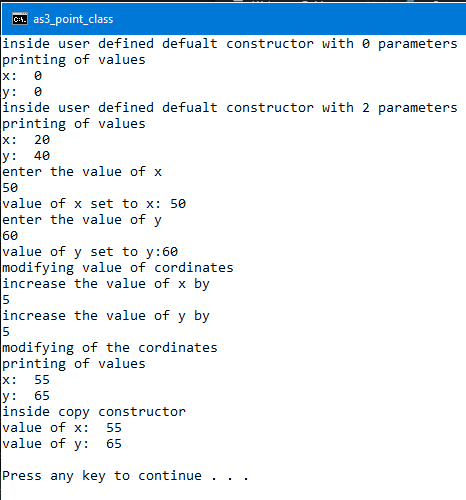
t1.modify(a,b);

t1.print();

tpoint c2(t1);//call to user defined copy constructor to copy t1 object

return 0;

}



**Q-wap to create a employee class that takes employee id, name, department name and show the functioning of different constructors, access modifiers, copy constructor**

**Ans-**

#include <iostream>

#include <stdlib.h>

#include <string.h>

#include <cstring>

using namespace std;

class emp

{

private:

int eid;

char ename[30];

char edept[10];

public:

void setvalues();//set employee value

void print();

emp();//e user defined defualt constructor setting defualt employee data

emp(int id, char name[30], char dept[10] );// e1 user defined default constructor to set employee id name deprtment;

emp(const emp &e0)

{

cout<<"inside copy constructor to copy employee\n";

eid=e0.eid;

strcpy(ename,e0.ename);

strcpy(edept,e0.edept);

print();

}

};

void emp::setvalues()

{

cout<<"enter employee id:";

cin>>eid;

cout<<"enter emplyee name:";

cin.getline(ename,30,'\n');

cout<<"enter employee department name:";

cin.getline(edept,10,'\n');

}

void emp::print()

{

cout<<"eployee id: "<<eid<<"\n";

cout << "employee name: "<<ename<< '\n';

cout << "employee department no: "<<edept<< '\n';

}

emp::emp()

{

cout << "user defined defualt constructor setting defualt employee data" << '\n';

eid=0;

strcpy(ename,"NULL");

strcpy(edept,"NULL");

print();

}

emp::emp(int id, char name[30], char dept[10] )

{

cout << "parameterised user defined default constructor to set employee id name deprtment" << '\n';

eid=id;

strcpy(ename,name);

strcpy(edept,dept);

print();

}

int main()

{

//int id;

//string name;

//float dept\_no;

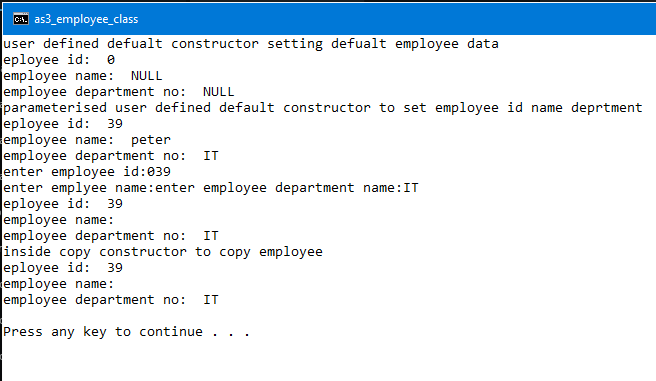
emp e;

emp e1(39,"peter","IT");

e.setvalues();

e.print();

emp e2(e);return 0;}



**Lab ASSIGMENT-4**

**Q-wap to create class point and show the functioning of constructor parameterized constructor, passing object to a function, returning object from a function,**

**Ans-**

#include <iostream>

#include<stdlib.h>

using namespace std;

class kpoint

{

private:

int x,y,z;

public:

kpoint();

kpoint(int a,int b,int c);

kpoint(const kpoint &k);

void add(const kpoint &k);

kpoint subtract(kpoint k);

void setvalues();

void print();

};

kpoint::kpoint()

{

cout<<"inside system default constructor\n";

x=0;

y=0;

z=0;

print();

}

kpoint::kpoint(int a,int b,int c)

{

cout<<"inside parameterized constructor\n";

x=a;

y=b;

z=c;

print();

}

kpoint::kpoint(const kpoint &k)

{

cout<<"inside default copy constructor\n";

x=k.x;

y=k.y;

z=k.z;

print();

}

void kpoint::add(const kpoint &k)

{

cout << "inside add function" << '\n';

cout<<"taking object as parameter\n";

x=x+k.x;

y=y+k.y;

z=z+k.z;

print();

}

kpoint kpoint::subtract(kpoint k)

{

cout << "inside subtract function" << '\n';

cout<<"taking object as parameter and returning object\n";

kpoint temp;

temp.x=x-k.x;

temp.y=y-k.y;

temp.z=z-k.z;

return temp;

}

void kpoint::setvalues()

{

cout << "enter the values of x: ";

cin>>x;

cout << "enter the values of y: ";

cin>>y;

cout << "enter the values of z: ";

cin>>z;

}

void kpoint::print()

{

cout << "printing of values" << '\n';

cout << "x:"<<x<<"\n";

cout << "y:"<<y<<"\n";

cout << "z:"<<z<<"\n";

}

int main()

{

kpoint k1;

k1.setvalues();

kpoint k2(10,20,30);

k1.add(k2);

k1.print();

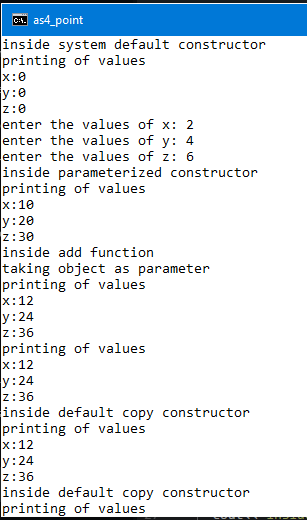
kpoint k3(k1);

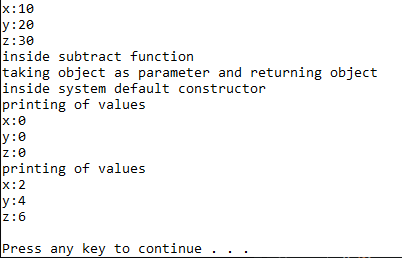
kpoint k4=k3.subtract(k2);

k4.print();

return 0;

}





**Q-wap to create fraction class and show the addition of numerator & functioning of constructor parameterized constructor, passing object to a function, returning object from a function**

**Ans-**

#include <iostream>

#include<stdlib.h>

using namespace std;

class kpoint

{

private:

int x,y,z;

public:

kpoint();

kpoint(int a,int b,int c);

kpoint(const kpoint &k);

void add(const kpoint &k);

kpoint subtract(kpoint k);

void setvalues();

void print();

};

kpoint::kpoint()

{

cout<<"inside system default constructor\n";

x=0;

y=0;

z=0;

print();

}

kpoint::kpoint(int a,int b,int c)

{

cout<<"inside parameterized constructor\n";

x=a;

y=b;

z=c;

print();

}

kpoint::kpoint(const kpoint &k)

{

cout<<"inside default copy constructor\n";

x=k.x;

y=k.y;

z=k.z;

print();

}

void kpoint::add(const kpoint &k)

{

cout << "inside add function" << '\n';

cout<<"taking object as parameter\n";

x=x+k.x;

y=y+k.y;

z=z+k.z;

print();

}

kpoint kpoint::subtract(kpoint k)

{

cout << "inside subtract function" << '\n';

cout<<"taking object as parameter and returning object\n";

kpoint temp;

temp.x=x-k.x;

temp.y=y-k.y;

temp.z=z-k.z;

return temp;

}

void kpoint::setvalues()

{

cout << "enter the values of x: ";

cin>>x;

cout << "enter the values of y: ";

cin>>y;

cout << "enter the values of z: ";

cin>>z;

}

void kpoint::print()

{

cout << "printing of values" << '\n';

cout << "x:"<<x<<"\n";

cout << "y:"<<y<<"\n";

cout << "z:"<<z<<"\n";

}

int main()

{

kpoint k1;

k1.setvalues();

kpoint k2(10,20,30);

k1.add(k2);

k1.print();

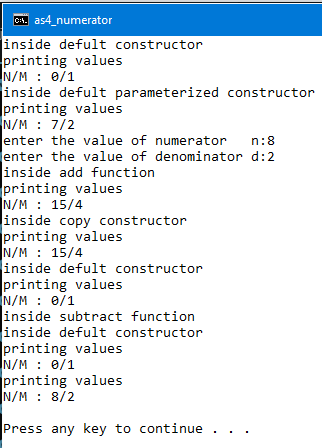
kpoint k3(k1);

kpoint k4=k3.subtract(k2);

k4.print();

return 0;

}



# Lab Assignment-5

**Q-wap to create a car class taking char pointers as a class members. Show the functioning of default constructor, parameterized constructor.**

**Ans-**

#include<iostream>

using namespace std;

#include<cstring>

#include<string.h>

class car

{

private:

int id;

char \*man;

char \*mod;

public:

car()

{

cout<<"inside defualt constructor\n";

id=0;

man=new char[strlen("unknown")+1];

strcpy(man,"unknown");

mod=new char[strlen("unknown")+1];

strcpy(mod,"unknown");

print();

}

car(int i,char ma[],char mo[])

{

cout<<"inside parameterised defualt constructor\n";

id=i;

man=new char[strlen(ma)];

strcpy(man,ma);

mod=new char[strlen(mo)];

strcpy(mod,mo);

print();

}

void print()

{

cout<<"car id is: "<<id<<"\n";

cout<<"manufacturer of car is: "<<man<<"\n";

cout << "car model is :" <<mod<< '\n';

}

};

int main()

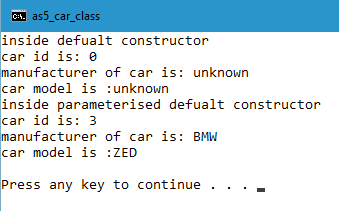
{

car c1;

car c2(03,"BMW","ZED");

return 0;

}



**Q- Wap to create a student class taking char & int pointers as class members.show the functioning of default constructor, parameterized constructor,destructor.**

**Ans-**

#include<iostream>

using namespace std;

#include<cstring>

#include<string.h>

class student

{

private:

int rollno;

char \*name;

int \*marks;

public:

student()

{

cout<<"inside defualt constructor\n";

rollno=0;

name=new char[strlen("unknown")+1];

strcpy(name,"unknown");

marks=new int(-1);

print();

cout<<"student marks: "<<\*marks<<"";

}

student(int r,char n[],int \*m)

{

//int \*p;

rollno=r;

name=new char[strlen(n)];

strcpy(name,n);

marks=new int[5];

print();

cout << "student marks: ";

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

{

\*(marks+i)=\*(m+i);

cout<<\*(marks+i)<<" ";

}

}

~ student()

{

cout<<"\n destructor called haha\n";

delete name;

delete marks;

}

void print()

{

cout<<"student roll no.: "<<rollno<<endl;

cout <<"student name :"<<name<< '\n';

}

};

int main()

{

student s1;

int m[]={90,80,81,93};

s1.~ student();

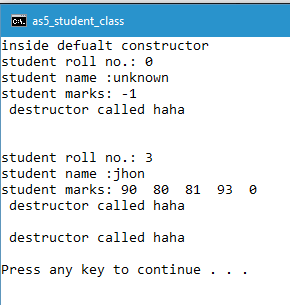
cout<<"\n\n";

student s2(3,"jhon",m);

s2.~ student();

return 0;

}



**Lab assignment-6**

**Q- wap to show the functioning of container class and component class. By creating two classes point class (component class) circle class (container class)**

**Ans-**

#include<stdio.h>

#include<iostream>

using namespace std;

class point //COMPONENT CLASS

{

//private:

public:

int x,y;

point()

{

cout<<"COMPONENT CLASS default constructor\n";

x=0;

y=0;

print();

}

point(int a,int b)

{

cout<<"COMPONENT CLASS parameterised default constructor\n";

x=a;

y=b;

print();

}

point(const point &p)

{

cout<<"COMPONENT CLASS copy constructor\n";

x=p.x;

y=p.y;

print();

}

void print()

{

//cout<<"POINT COMPONENT CLASS\n";

cout<<"point x: "<<x<<"\n";

cout<<"point y: "<<y<<"\n";

}

/\*void hello()

{

cout<<"hello from container class\n";

}\*/

};

class circle

{

private:

int radius;

point center;

public:

circle()

:center()

{

radius=0;

cout<<"CONTAINER CLASS default constructor\n";

cprint();

}

circle(int x,int y,int z)

:center(y,z)

{

cout<<"CONTAINER CLASS parameterized constructor\n";

radius=x;

cprint();

}

circle(point &p,int r)

:center(p)

{

cout<<"CONTAINER CLASS constructor\n";

radius=r;

cprint();

}

void cprint()

{

cout <<"radius of circle: "<<radius<< '\n';

cout <<"center of circle: "<<center.x<<" "<<center.y<< '\n';

}

};

int main()

{

circle c1;

cout<<"\n\n";

circle c2(10,2,2);

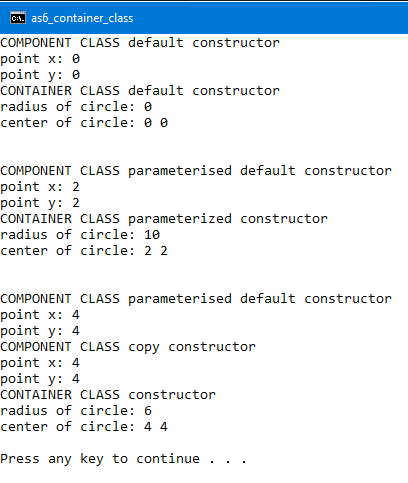
cout<<"\n\n";

point p1(4,4);

circle c3(p1,6);

return 0;

}



**Q- wap to show the functioning of container class and component class and working of system defined constructor to print garbage value . By creating two classes point class (component class) circle class (container class)**

**Ans-**

#include <iostream>

#include<stdio.h>

using namespace std;

class point

{

public:

int x,y;

void print()

{

cout<<"point x: "<<x<<"\n";

cout<<"point y: "<<y<<"\n";

}

};

class circle

{

int radius;

point center;

public:

void print()

{

cout <<"radius of circle: "<<radius<< '\n';

cout <<"center of circle: "<<center.x<<" "<<center.y<< '\n';

}

};

int main()

{

cout << "calling of system default constructor of component & container class" << '\n';

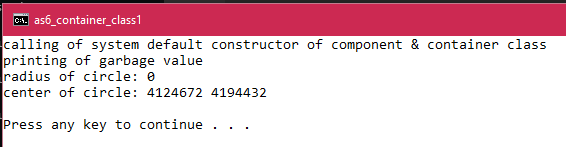
cout <<"printing of garbage value\n";

circle c1;

c1.print();

return 0;

}



**Q- wap to show the functioning of container class and component class also take pointers as class members . By creating three classes author class, publisher class (component class) book class (container class) also create default constructor, parameterized constructor, copy constructor.**

**Ans-**

#include<stdio.h>

#include<cstring>

#include<iostream>

using namespace std;

class author

{

private:

char \*aname;

char \*aadd;

public:

author()

{

cout<<"default constructor author class\n";

aname=new char[strlen("unknown")+1];

aadd=new char[strlen("unknown")+1];

strcpy(aname,"unlnown");

strcpy(aadd,"unknown");

aprint();

}

author(char an[],char ad[])

{

cout<<"parameterised constructor author class\n";

aname=new char[strlen(an)];

aadd=new char[strlen(ad)];

strcpy(aname,an);

strcpy(aadd,ad);

aprint();

}

author(const author &a)

{

cout<<"default copy constructor author class\n";

aname=a.aname;

aadd=a.aadd;

aprint();

}

void aprint()

{

cout<<"author name: "<<aname<<endl;

cout<<"author address: "<<aadd<<endl;

}

};

class publisher

{

private:

char \*pname;

char \*padd;

public:

publisher()

{

cout<<"default constructor publisher class\n";

pname=new char[strlen("unknown")+1];

padd=new char[strlen("unknown")+1];

strcpy(pname,"unknown");

strcpy(padd,"unknown");

pprint();

}

publisher(char pn[],char pa[])

{

cout<<"parameterised constructor publisher class\n";

pname=new char[strlen(pn)];

padd=new char[strlen(pa)+1];

strcpy(pname,pn);

strcpy(padd,pa);

pprint();

}

publisher(const publisher &p)

{

cout<<"copy constructor publisher class\n";

pname=p.pname;

padd=p.padd;

pprint();

}

void pprint()

{

cout<<"publisher name: "<<pname<<endl;

cout<<"publisher address: "<<padd<<endl;

}

};

class book

{

private:

int bid;

char \*bname;

author a1;

publisher p1;

public:

book():

a1(),p1()

{

cout<<"default constructor book class\n";

bid=0;

bname=new char[strlen("unknown"+1)];

strcpy(bname,"unknown");

bprint();

}

book(char an[],char ad[],char pn[],char pd[],char bn[],int id):

a1(an,ad),p1(pn,pd)

{

cout<<"parameterised constructor book class\n";

bid=id;

bname=new char[strlen(bn)];

strcpy(bname,bn);

bprint();

}

book(author &a,publisher &p,char bn[],int id):

a1(a),p1(p)

{

cout<<"copy constructor book class\n";

bid=id;

bname=new char[strlen(bn)];

strcpy(bname,bn);

bprint();

}

void bprint()

{

cout<<"book id: "<<bid<<endl;

cout<<"book name: "<<bname<<endl;

}

};

int main()

{

cout<<"\n\n";

cout<<"calling parameterized component class constructor\n";

author a("sgt price","los angeles");

publisher p("lootcrate","miami");

cout<<"\n\n";

cout<<"calling default component class constructor from container class\n";

book b1;

cout<<"\n\n";

cout<<"calling parameterized component class constructor from container class\n";

book b2("weekend","denver","lady","colarado","monster",96);

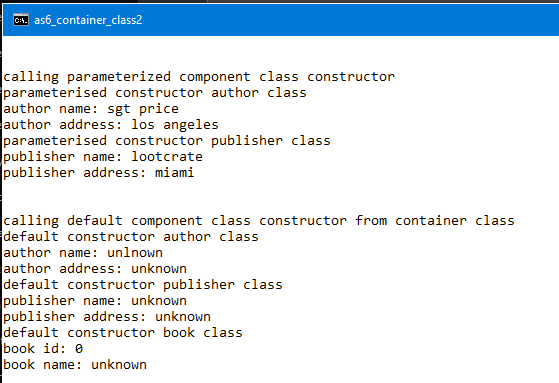
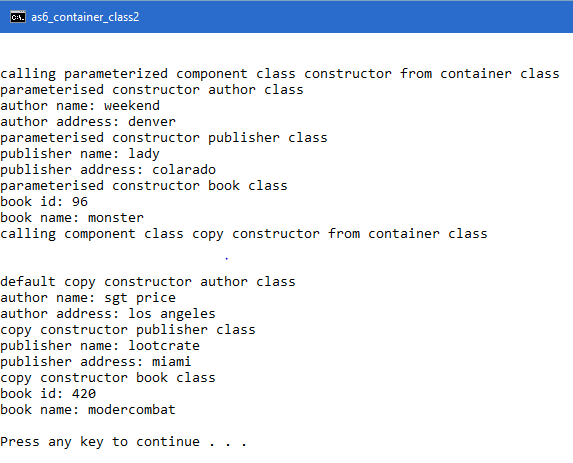
cout<<"calling component class copy constructor from container class\n";

cout<<"\n\n";

book b3(a,p,"modercombat",420);

return 0;

}



**Lab assignment-7**

**Q- wap to show the functioning of dynamic object by creation of container class and component class.**

**Ans-**

#include<iostream>

#include<stdio.h>

#include<cstring>

using namespace std;

class rider

{

private:

char \*rname;

char \*rcity;

int rage;

public:

rider()

{

cout<<"rider default constructor\n";

rname=new char[strlen("unknown")+1];

rcity=new char[strlen("unknown")+1];

strcpy(rname,"unknown");

strcpy(rcity,"unknown");

rage=0;

}

rider(char name[],char city[],int age)

{

cout<<"rider parameterized constructor\n";

rname=new char[strlen(name)];

rcity=new char[strlen(city)];

strcpy(rname,name);

strcpy(rcity,city);

rage=age;

}

rider(const rider &r)

{

cout<<"rider copy constructor\n";

rname=r.rname;

rcity=r.rcity;

rage=r.rage;

}

void rprint()

{

cout<<"rider name is: "<<rname<<endl;

cout<<"rider age is: "<<rage<<endl;

cout<<"rider city is: "<<rcity<<endl;

}

};

class wheel

{

private:

int wid;

char \*wname;

public:

wheel()

{

cout<<"wheel class default constructor\n";

wname=new char [strlen("unknown")+1];

strcpy(wname,"unknown");

wid=0;

}

wheel(int id,char name[])

{

cout<<"wheel class parameterized constructor\n";

wid=id;

wname=new char [strlen(name)+1];

strcpy(wname,name);

}

wheel(const wheel &w)

{

cout<<"wheel class copy constructor\n";

wid=w.wid;

wname=w.wname;

}

void wprint()

{

cout<<"wheels id: "<<wid<<endl;

cout<<"wheel manufacturer is:"<<wname<<endl;

}

};

class bike

{

private:

char \*bname;

int bid;

wheel \*wo;

rider \*ro;

public:

bike(char name[],int id,wheel &wc,rider &rc)

{

cout<<"inside bike container class\n";

cout<<"complete information\n";

ro=new rider(rc);

bname=new char[strlen(name)];

strcpy(bname,name);

bid=id;

wo=new wheel(wc);

}

void bprint()

{

ro->rprint();

cout<<"bike id is: "<<bid<<endl;

cout<<"riders bike name: "<<bname<<endl;

wo->wprint();

delete wo;

delete ro;

}

};

int main()

{

wheel \*w1=new wheel(39,"ceat");

w1->wprint();

cout<<"\n\n";

rider \*r1=new rider("xenos","nirvana",18);

r1->rprint();

cout<<"\n\n";

bike \*b1=new bike("FURY",20,\*w1,\*r1);

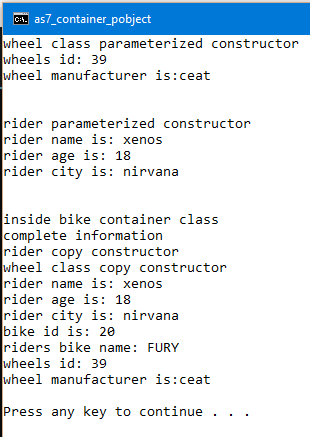
b1->bprint();

delete b1;

delete w1;

delete r1;

}



**Q- wap to show the functioning of dynamic object by creation of bike wheel and rider class where rider class has a dynamic object .**

#include<iostream>

using namespace std;

#include <cstring>

#include <stdio.h>

class rider

{

private:

char \*rname;

int rage;

char \*rcity;

public:

rider()

{

rname=new char[strlen("empty")+1];

strcpy(rname,"empty");

rage=0;

rcity=new char[strlen("NULL")+1];

strcpy(rcity,"NULL");

rprint();

}

rider(char rn[],int ra,char rc[])

{

rname=new char[strlen(rn)];

strcpy(rname,rn);

rage=ra;

rcity=new char[strlen(rc)];

strcpy(rcity,rc);

rprint();

}

rider(const rider &ro)

{

rname=new char[strlen(ro.rname)];

strcpy(rname,ro.rname);

rage=ro.rage;

rcity=new char[strlen(ro.rcity)];

strcpy(rcity,ro.rcity);

rprint();

}

void rprint()

{

cout<<"riders name:"<<rname<<endl;

cout<<"riders age:"<<rage<<endl;

cout<<"riders city:"<<rcity<<endl;

}

};

class wheel

{

private:

char \*wman;

int wprice;

public:

wheel()

{

wman=new char[strlen("NULL")+1];

strcpy(wman,"NULL");

wprice=0;

wprint();

}

wheel(char wm[],int wp)

{

wman=new char[strlen(wm)+1];

strcpy(wman,wm);

wprice=wp;

}

wheel(const wheel &wo)

{

wman=new char[strlen(wo.wman)];

strcpy(wman,wo.wman);

wprice=wo.wprice;

wprint();

}

void wprint()

{

cout<<"wheels manufacturer: "<<wman<<endl;

cout<<"wheels price: "<<wprice<<endl;

}

};

class bike

{

private:

int bikeid;

char \*bname;

wheel w1;

rider \*r1;

public:

bike()

:w1()

{

bname=new char[strlen("no bike")+1];

strcpy(bname,"no bike");

bikeid=0;

bprint();

}

bike(int bi,char bn[],char wn[],int wp)

:w1(wn,wp)

{

bname=new char[strlen(bn)+1];

strcpy(bname,bn);

bikeid=bi;

bprint();

}

void bprint()

{

cout<<"bike name: "<<bname<<endl;

cout<<"bike id: "<<bikeid<<endl;

w1.wprint();

}

void rideron()

{

char rn[10];

int ra=0;

char rc[10];

cout<<"eneter riders name: ";

cin>>rn;

cout<<"enter riders age:";

cin>>ra;

cout<<"enter riders city:";

cin>>rc;

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

r1=new rider(rn,ra,rc);

}

void riderof()

{

delete r1;

r1=new rider();

}

};

int main()

{

char ch='y';

bike b1(39,"pulsar","mrf",2000);

cout<<"assign bike to rider\n";

while(ch=='y')

{

cout<<"press y to assign & n to dismount rider\n";

cin>>ch;

if(ch=='y'||ch=='Y')

{

b1.rideron();

b1.bprint();

}

else

{

b1.riderof();

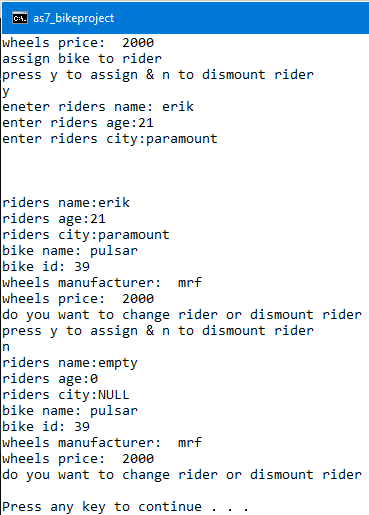
b1.bprint();

}

cout<<"do you want to change rider or dismount rider"<<endl;

}

}



**Lab assignment-8**

**Q- wap to show the functioning of operator overloading, show addition subtraction and printing of object with the help of operator.**

#include<iostream>

#include<cstring>

using namespace std;

class fraction

{

private :

int nem;

int dem;

public :

fraction()

{

cout<<"setting of default values\n";

nem=0;

dem=0;

}

fraction(int a, int b)

{

cout<<"setting of values\n";

nem=a;

dem=b;

fprint();

}

void operator ==(fraction &f)

{

if(nem==f.nem&&dem==f.dem)

{

cout<<"fractions are equal\n";

}

else

{

cout<<"fraction are not equal\n";

}

if(dem==f.dem)

{

cout<<"they are like fraction \n";

}

else

{

cout<<"they are unlike fraction";

}

}

void operator <<(ostream &out)

{

out<<nem<<"/"<<dem;

}

void operator <(fraction &f)

{

float r1,r2;

r1=nem/dem;

r2=f.nem/f.dem;

if(r1<r2)

{

cout<<nem<<"/"<<dem<<": is smaller than"<<f.nem<<"/"<<f.dem<<endl;

}

else

{

cout<<nem<<"/"<<dem<<": is greater than"<<f.nem<<"/"<<f.dem<<endl;

}

}

fprint()

{

cout<<"="<<nem<<"/"<<dem<<endl;

}

};

int main()

{

cout<<"f1:";

fraction f1(2,7);

cout<<"f2:";

fraction f2(2,8);

f1==f2;

f1<f2;

cout<<"printing of object\n";

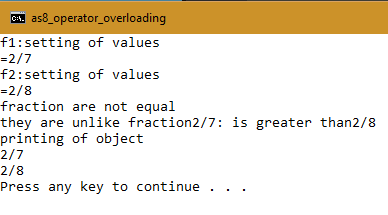
f1<<cout;

cout<<endl;

f2<<cout;

return 0;

}



**Q- wap to show the functioning of operator overloading ?**

#include<iostream>

using namespace std;

#include<stdio.h>

#include<stdlib.h>

class point

{

private:

int x,y;

public:

point()

{

x=y=0;

}

point(int a,int b)

{

cout<<"creating the point\n";

x=a;

y=b;

display();

}

istream& operator>>(istream &in)

{

cout<<"cin operation\n";

in>>x;

if(in.ios::bad()!=0)

{

return in;

}

in>>y;

if(in.ios::bad()!=0)

{

return in;

}

return in;

}

point& operator=(point &p)

{

cout<<"assigment operator\n";

x=p.x;

y=p.y;

return \*this;

}

void operator ++()

{

cout<<"post increment operator\n";

x=x+1;

y=y+1;

}

point operator ++(int)

{

cout << "post increment operator" << '\n';

point temp;

temp.x=x+1;

temp.y=y+1;

return temp;

}

void display()

{

cout<<"the 2 points are"<<endl;

cout<<"x: "<<x<<endl;

cout<<"y: "<<y<<endl;

}

};

int main()

{

int ch;

cout<<"1. cin >> operator"<<endl;

cout<<"2. = assigment operator"<<endl;

cout<<"3. ++o pre increment operator "<<endl;

cout<<"4. o++ post increment operator "<<endl;

cout<<"5. exit "<<endl;

while(1)

{

cout << "select your option\n" <<endl;

cin>>ch;

switch (ch)

{

case 1:

{

point p1;

point p2;

p1>>(p2>>cin);

p1.display();

p2.display();

break;

}

case 2:

{

point p3(3,4);

point p4(1,2);

p3=p4;

p3.display();

p4.display();

break;

}

case 3:

{

point p3(1,2);

++p3;

p3.display();

break;

}

case 4:

{

point p3(1,2);

p3++;

p3.display();

break;

}

case 5:

{

exit(0);

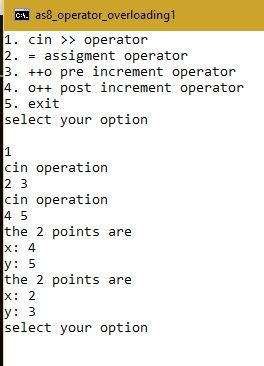
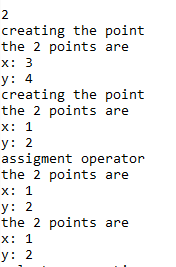
}

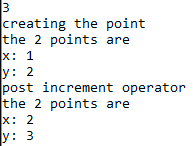
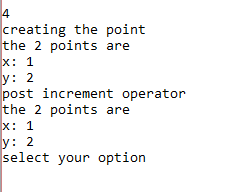
}

}

return 0;

}

**Q- wap to show the functioning of stack ?**

#include"iostream"

#include<conio.h>

#define size 5

using namespace std;

class stack

{

int \*a;

int top;

public:

stack()

{

top=-1;

a=new int[size];

}

int isFull()

{

if(top==size-1)

return 1;

else

return 0;

}

int isEmpty()

{

if(top==-1)

return 1;

else

return 0;

}

void push(int value)

{

cout<<"Value inserted in Stack :"<<value<<"\n";

if(this->isFull())

cout<<"Stack is Full\n";

else

a[++top]=value;

}

void pop()

{

cout<<"\nValue deleted from Stack :";

if(this->isEmpty())

cout<<"Stack is Empty\n";

else

{

cout<<a[top--];

cout<<endl;

}

}

void display()

{

cout<<"\nStack : ";

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

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

}

};

int main()

{

stack s;

s.push(6);

s.push(3);

s.push(4);

s.display();

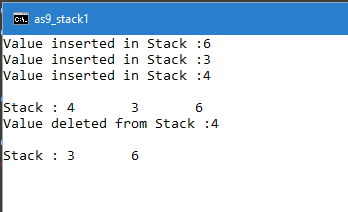
s.pop();

s.display();

getch();

return 0;

}



**Lab Assigment-9**

**Q- wap to show the operator overloading of sub-script operator and generate random numbers ?**

#include<iostream>

using namespace std;

#include<cstdlib>

class inset

{

private:

int size;

int \*value;

public:

inset(int s)

{

size=s;

value=new int[4];

}

int& operator[] (int index)

{

if(index>size)

{

cout<<"index out of bound\n";

}

return value[index];

}

};

void display(inset &is,int size)

{

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

{

cout << is[i] <<endl;

}

}

int main()

{

int s;

cout<<"program to print random no.\n";

cout<<"enter the no. of random no. you want to genrate"<<endl;

cin>>s;

inset i1(s);

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

{

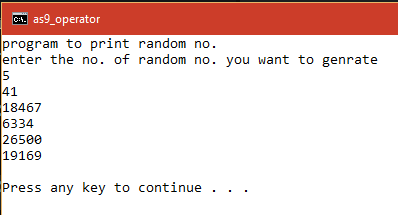
int num=rand();

i1[i]=num;

}

display(i1,s);

}



**Q- wap to show the operator overloading of sub-script operator and add student marks to an array ?**

#include<iostream>

using namespace std;

class Intset{

int Size, \*value;

public:

Intset(){

Size=0;

value=NULL;

}

Intset(int s)

{

Size = s;

value = new int[Size];

}

int& operator[](int index)

{

return value[index];

}

void display(){

cout<<"\nMarks of the subjects are: ";

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

{

cout<<\*(value+i)<<"\t";

}

}

};

int main(void){

cout<<"\nEnter the number of subjects";

int subs;

cin>>subs;

Intset i1(subs);

cout<<"\nEnter the marks for each subjects";

int marks;

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

cout<<"\nEnter marks: ";

cin>>marks;

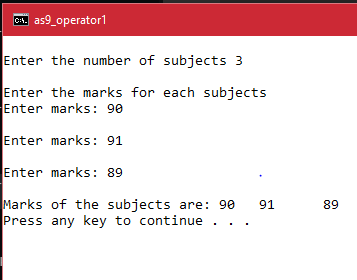
i1[i] = marks;

}

i1.display();

return 0;

}



**Lab Assigment-10**

**Q- wap to create currency converter and convert INR currency to USD currency and vice versa ?**

**Ans-**

#include <iostream>

using namespace std;

#include <stdlib.h>

void check(int &dr,int &pc)

{

while(pc>=100)

{

dr=dr+1;

pc=pc-100;

}

}

class usa

{

private:

int dollar;

int cents;

public:

usa()

{

dollar=cents=0;

}

usa(int d,int c)

{

dollar=d;

cents=c;

}

int get\_dollar()

{

return dollar;

}

int get\_cents()

{

return cents;

}

void set\_uvalues(int d, int c)

{

check(d,c);

dollar=d;

cents=c;

}

void udisplay()

{

cout<<dollar<<": dollar "<<cents<<": cents"<<endl;

}

};

class india

{

private:

int rupees;

int paisa;

public:

india()

{

rupees=paisa=0;

}

india(int r,int p)

{

rupees=r;

paisa=p;

}

int get\_rupees()

{

return rupees;

}

int get\_paisa()

{

return paisa;

}

void set\_ivalues(int r,int p)

{

check(r,p);

rupees=r;

paisa=p;

}

void idisplay()

{

cout<<rupees<<":rupees "<<paisa<<":paisa "<<endl;

}

};

void convert2r(india &ic, usa &uc)

{

int nr,np;

nr=64\*uc.get\_dollar();

np=64\*uc.get\_cents();

ic.set\_ivalues(nr,np);

ic.idisplay();

}

void convert2d(india &ic, usa &uc)

{

int nd,nc;

nd=ic.get\_rupees()/64;

nc=ic.get\_paisa()/64;

uc.set\_uvalues(nd,nc);

uc.udisplay();

}

int main()

{

int ch;

cout<<"welcome to curency converter\n";

while (1) {

cout<<"1. convert INR to USD"<<endl;

cout<<"2. convert USD to INR"<<endl;

cout<<"3.exit\n";

cout<<"enter your choice\n";

cin>>ch;

switch (ch) {

case 1:

{

int rs,ps;

cout<<"INR TO USD\n";

cout<<"enter rupees you want to convert\n";

cin>>rs;

cout<<"enter the paisa you want to convert\n";

cin>>ps;

india i(rs,ps);

i.idisplay();

usa u;

convert2d(i,u);

cout << "\n\n";

break;

}

case 2:

{

int ds,cs;

cout<<"USD TO INR\n";

cout<<"enter dollars you want to convert\n";

cin>>ds;

cout<<"enter the cents you want to convert\n";

cin>>cs;

usa u(ds,cs);

u.udisplay();

india i;

convert2r(i,u);

cout << "\n\n";

break;

}

case 3:

{

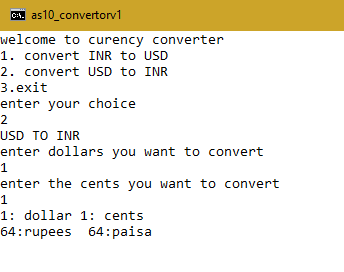
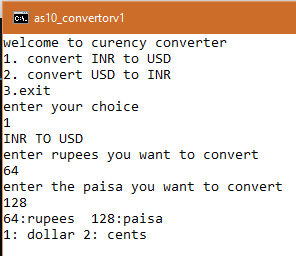
exit(0);

}

}

}

}



**Q- wap to create metric converter and convert foot/inches to meter/centimeter and vice versa ?**

#include<iostream>

using namespace std;

#include <stdlib.h>

class FI

{

private:

float foot;

float inches;

public:

FI()

{

foot=0.0;

inches=0.0;

}

FI(float f, float i)

{

foot=f;

inches=i;

}

float get\_foot()

{

return foot;

}

float get\_inches()

{

return inches;

}

void set\_fivalues(float f,float i)

{

foot=f;

inches=i;

}

void fidisplay()

{

cout<<foot<<":foot "<<inches<<":inches "<<endl;

cout<<"\n";

}

};

class MC

{

private:

float meter;

float centimeter;

public:

MC()

{

meter=0;

centimeter=0;

}

MC(float m, float c)

{

meter=m;

centimeter=c;

}

float get\_meter()

{

return meter;

}

int get\_centimeter()

{

return centimeter;

}

void set\_mcvalues(float m,float c)

{

meter=m;

centimeter=c;

}

void mcdisplay()

{

cout<<meter<<":meter "<<centimeter<<":in "<<endl;

cout<<"\n";

}

};

void fi2mc(FI &f1)

{

float m,c;

m=f1.get\_foot()\*0.30;

c=f1.get\_inches()\*2.45;

while(c>=100)

{

m+=1;

c=c-100;

}

MC m1(m,c);

m1.mcdisplay();

}

void mc2fi(MC &m1)

{

float f,i;

f=m1.get\_meter()\*3.2;

i=m1.get\_centimeter()\*0.30;

while(i>=12)

{

f+=1;

i=i-12;

}

FI f1(f,i);

f1.fidisplay();

}

int main()

{

cout<<"1 foot = 12 inches"<<endl;

cout<<"1 foot = 0.30 meter"<<endl;

cout<<"1 meter = 100 cm "<<endl;

cout<<"1 meter = 3.2 feet"<<endl;

cout<<"1 centimeter=0.30 inches"<<endl;

cout<<"1 inches=2.45 centimeter"<<endl;

int ch;

cout<<"welcome to metric conversion\n";

while (1) {

cout<<"1. convert foot/inches to meter/centimeter"<<endl;

cout<<"2. convert meter/centimeter to foot/inches"<<endl;

cout<<"3.exit\n";

cout<<"enter your choice\n";

cin>>ch;

switch (ch) {

case 1:

{

float f,i;

cout<<"foot/inches to meter/centimeter\n";

cout<<"enter foot you want to convert\n";

cin>>f;

cout<<"enter the inches you want to convert\n";

cin>>i;

FI fi(f,i);

fi.fidisplay();

fi2mc(fi);

cout << "\n\n";

break;

}

case 2:

{

float m,c;

cout<<"meter/centimeter to foot/inches\n";

cout<<"enter meter you want to convert\n";

cin>>m;

cout<<"enter the centimeter\n";

cin>>c;

MC mc(m,c);

mc.mcdisplay();

mc2fi(mc);

cout << "\n\n";

break; }

case 3:

{

exit(0);

}

}

}

}

**Q- wap to create a employee class and manager class and show the basic functioning of inheritance**

**Ans-**

#include<iostream>

#include<cstring>

#include<stdlib.h>

using namespace std;

class employee

{

private:

char \*ename;

int eno;

public:

employee()

{

cout<<"default constructor\n";

ename=new char[strlen("newbie")+1];

strcpy(ename,"newbie");

eno=0;

eprint();

}

employee(char name[10],int eid)

{

cout<<"parameterized constructor\n";

ename=new char[strlen(name)+1];

strcpy(ename,name);

eno=eid;

eprint();

}

employee(const employee &e1)

{

cout<<"copy constructor\n";

ename=new char[strlen(e1.ename)+1];

strcpy(ename,e1.ename);

eno=e1.eno;

eprint();

}

void eprint()

{

cout<<"employee name:"<<ename<<endl;

cout<<"employee id:"<<eno<<endl;

}

};

class manager:public employee

{

private:

int dno;

public:

manager():employee()

{

dno=0;

mprint();

}

manager(char name[],int eid,int did):employee(name,eid)

{

dno=did;

mprint();

}

manager(const manager &m1):employee(m1)

{

dno=m1.dno;

mprint();

}

void mprint()

{

cout<<"department id:"<<dno<<endl;

}

};

int main()

{

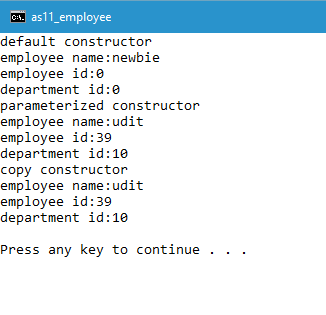
manager m;

manager m0("udit",39,10);

manager m2=m0;

return 0;

}



**Q- wap to create a 2point class and 3point class and show the basic functioning of inheritance**

**Ans-**

#include<iostream>

using namespace std;

class point

{

private:

int a,b;

public:

point()

{

cout<<"default constructor\n";

a=0;

b=0;

ppoint();

}

point(int x, int y)

{

cout<<"parameterized constructor\n";

a=x;

b=y;

ppoint();

}

point(const point &p1)

{

cout<<"copy constructor\n";

a=p1.a;

b=p1.b;

ppoint();

}

void ppoint()

{

cout<<"a:"<<a<<endl<<"b:"<<b<<endl;

}

};

class tdpoint:public point

{

private:

int c;

public:

tdpoint():point()

{

c=0;

ttdpoint();

}

tdpoint(int x, int y, int z):point(x,y)

{

c=z;

ttdpoint();

}

tdpoint(const tdpoint &t1):point(t1)

{

c=t1.c;

ttdpoint();

}

void ttdpoint()

{

cout<<"c:"<<c<<endl;

}

};

int main()

{

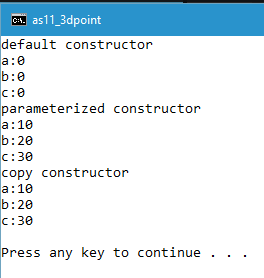
tdpoint t1 ;

tdpoint t2(10,20,30);

tdpoint t3=t2;

return 0;

}



**Q- wap to show the basic functioning of Access Specifier**

**Ans-**

#include"iostream"

#include<conio.h>

using namespace std;

class A

{

public: int a;

A()

{

a=10;

b=6;

c=8;

}

void print()

{

cout<<a<<"\t"<<b<<"\t"<<c<<endl;

}

protected: int b;

private: int c;

};

class B:public A

{

public:void bprint()

{

cout<<a<<"\t"<<b<<"\t"<<endl;

//cout<<"c is not accesible \n";

}

};

class C:protected A

{

public:void cprint()

{

cout<<a<<"\t"<<b<<"\t"<<endl;

}

};

class D:private A

{

public:void dprint()

{

cout<<a<<"\t"<<b<<"\t"<<endl;

}

};

int main()

{

A a1;

B b1;

C c1;

D d1;

a1.print();

b1.bprint();

c1.cprint();

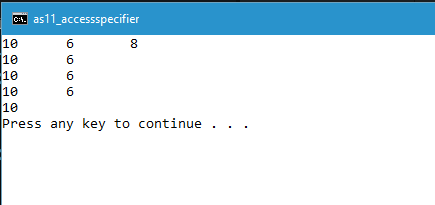
d1.dprint();

cout<<a1.a<<"\t";

return 0;

getch();

}



**Lab Assigment-12**

**Q- wap to show the basic functioning of calling of function by reference to an Object.**

**Ans-**

#include<iostream>

using namespace std;

#include<cstring>

class base

{

public:

void show()

{

cout<<"this is base class";

}

};

class derive: public base

{

public:

void show()

{

cout<<"this is derive class";

}

};

int main()

{

base \*b;

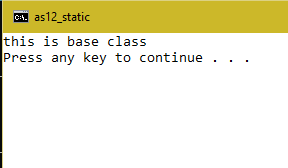
derive d;

b=&d;

b->show();

return 0;

}



**Q- wap to show the basic functioning of Virtual Function.**

**Ans-**

#include<iostream>

using namespace std;

#include<cstring>

class base

{

public:

virtual void show()

{

cout<<"this is base class";

}

};

class derive: public base

{

public:

void show()

{

cout<<"this is derive class";

}

};

int main()

{

base \*b;

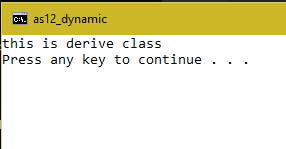
derive d;

b=&d;

b->show();

return 0;

}



**Q- wap to show the functioning of Hybrid Inheritance.**

**Ans-**

#include<iostream>

using namespace std;

#include<string.h>

class person

{

protected:

char \*pname;

int age;

public:

person(char name[],int a)

{

pname=new char[strlen(name)+1];

strcpy(pname,name);

age=a;

}

void print()

{

cout<<"name: "<<pname<<endl;

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

}

};

class teacher : virtual public person

{

protected:

char \*tcourse;

public:

teacher(char name[], int a,char course[]):person(name,a)

{

tcourse=new char[strlen(course)+1];

strcpy(tcourse,course);

}

void print()

{

cout<<"teaches: "<<tcourse<<endl;

}

};

class student: virtual public person

{

protected:

char \*ssubject;

public:

student(char name[], int a,char subject[]):person(name,a)

{

ssubject= new char[strlen(subject)+1];

strcpy(ssubject,subject);

}

void print()

{

cout<<"studies:"<<ssubject<<endl;

}

};

class phd: public teacher, public student

{

private:

char \*spec;

public:

phd(char name[], int a,char course[],char subject[],char speca[])

:person(name,a),teacher(name,a,course),student(name,a,subject)

{

spec=new char[strlen(speca)+1];

strcpy(spec,speca);

}

void print()

{

person::print();

teacher::print();

student::print();

cout<<"specalization:"<<spec;

}

} ;

int main()

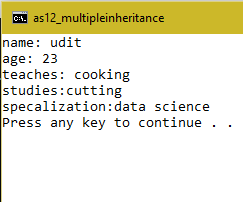
{

phd p1("udit",23,"xyz","mno","data science");

p1.print();

return 0;

}



**Q- wap to show the functioning Virtual Function.**

**Ans-**

#include<iostream>

using namespace std;

#include<cstring>

class gfigure

{

protected:

int gside;

public:

gfigure(int s)

{

gside=s;

}

virtual int area()=0;

virtual int perimeter()=0;

};

class rectangle: public gfigure

{

private:

int rside;

public:

rectangle(int l ,int b):gfigure(l)

{

rside=b;

}

int area()

{

int a=rside\*gside;

return a;

}

int perimeter()

{

int p=2\*(rside+gside);

return p;

}

};

int main()

{

int length,breadth,ar=0,pr=0;

cout<<"enter the length of rectangle \n";

cin>>length;

cout<<"enter the breadth of rectangle \n";

cin>>breadth;

gfigure \*g1;

rectangle r1(length,breadth);

g1=&r1;

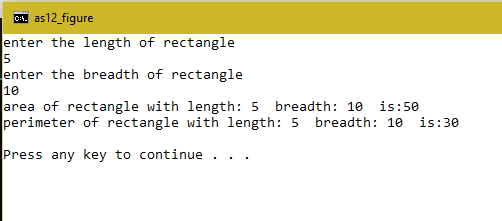
ar=g1->area();

cout<<"area of rectangle with length: "<<length<<" breadth: "<<breadth<<" is:"<<ar<<endl;

pr=g1->perimeter();

cout<<"perimeter of rectangle with length: "<<length<<" breadth: "<<breadth<<" is:"<<pr<<endl;

}



**Q- wap to show the functioning Exception Handling.**

**Ans-**

#include<iostream>

#include<stdlib.h>

using namespace std;

#define min 500

class account

{

private:

int balance;

public:

class low{};

account(int b)

{

cout<<"minimum balance should be: "<< min <<endl;

balance=b;

print();

}

void dep(int amount)

{

balance=balance+amount;

}

void wid(int amount)

{

int b=balance-amount;

if(b<min)

{

throw low();

}

else

{

balance=balance-amount;

}

print();

}

void print()

{

cout<<"current balance is:"<<balance<<endl;

}

};

int main()

{

account a1(600);

cout<<"enter the amount to withdraw\n";

int amount;

cin>>amount;

try

{

a1.wid(amount);

}

catch (account::low)

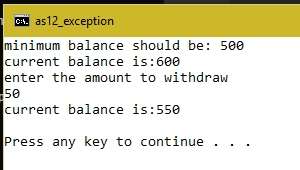
{

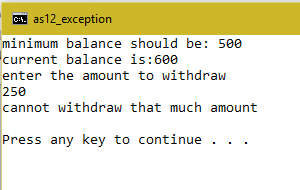
cout<<"cannot withdraw that much amount \n";

}

return 0;

}





**Lab Assigment-13**

Q-wap to show the functioning of standard library function(stl-0) ?

Ans-

#include<iostream>

using namespace std;

#include <algorithm>

#include <bits/stdc++.h>

void my\_fun(int a)

{

cout<<"element:"<<a\*10<<endl;

}

int my\_fun1(int a)

{

return (a\*100) ;

}

int main()

{

int arr[5]={10,11,22,11,40};

int \*p;

int n= sizeof(arr)/sizeof(arr[0]);

cout<<"first array\n";

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

{

cout<<\*(arr+i)<<"\t";

}

cout<<endl;

cout<<endl;

cout<<"find() stl function \n";

p=find(arr,arr+n,22);

cout<<"element 22 found at:"<<(p)-(arr)<<endl;

cout<<endl;

cout<<"count() stl function \n";

cout<<"no. of times 11 apear in array:"<<count(arr,arr+n,11)<<endl;

cout<<endl;

cout<<"for\_each() stl function // access each element of array for some operation pass each element to defined function \n";

for\_each(arr,arr+n,my\_fun);

cout<<endl;

cout<<"transform() stl function // access each element of array for some operation pass each element to defined function store modified value on new array\n";

int arr1[n];

std::transform(arr,arr+n,arr1,my\_fun1);

cout<<"second array after transform stl function \n";

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

{

cout<<\*(arr1+i)<<"\t";

}

cout<<endl;

cout<<"sort() & merge() stl function"<<endl;

int arr2[2\*n];

sort(arr,arr+n);

sort(arr1,arr1+n);

merge(arr,arr+n,arr1,arr1+n,arr2);

cout<<endl;

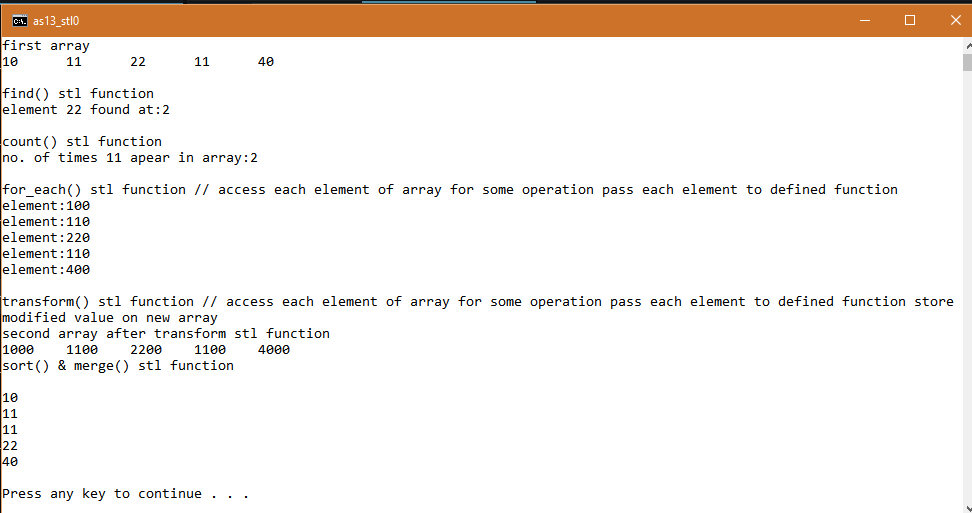
for(int i=0; i<n; i++)

{

cout<<\*(arr2+i)<<endl;

}

}



Q- Q-wap to show the functioning of standard library function(stl-1) list/vector/deque ?

Ans-

#include<iostream>

#include<list>

#include<vector>

#include<deque>

#include<algorithm>

#include "conio.h"

using namespace std;

int main()

{

int i;

while(1)

{

cout<<endl;

cout<<"1. vector operation\n";

cout<<"2. list operation\n";

cout<<"3. deque operation\n";

cout<<"4. more list operation\n";

cout<<"5. exit\n";

cin>>i;

switch(i)

{

case 1:

{

cout<<"vector operation\n";

cout<<"swap of two vector \n";

cout<<"traverse using pop\_back() \n";

double arr[]={10.1,10.2,10.3,10.4,10.5};

int n=sizeof(arr)/sizeof(arr[0]);

vector<double> v1(arr,arr+n);

cout<<"size of vector v1 is:"<<v1.size()<<endl;

cout<<"elements of vector v1 is:"<<endl;

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

{

cout<<v1[i]<<"\t";

}

cout<<endl;

vector<double> v2(n);

cout<<"swap(),pop\_back(),push\_back(),back(),size() stl vector function"<<endl;

v1.swap(v2);

cout<<"size of vector v2 is:"<<v2.size()<<endl;

cout<<"elements of vector v1 is:"<<endl;

while(!v2.empty())

{

cout<<v2.back()<<"\t";

v2.pop\_back();

}

cout<<endl;

cout<<endl;

}

break;

case 2:

{

cout<<"list operation\n";

cout<<"push\_front(),front(),pop\_front(),size() stl list functions\n";

int a[]={10,20,30,40,50};

int n=sizeof(a)/sizeof(a[0]);

list<int> li;

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

{

li.push\_front(a[i]);

}

cout<<"size of list li is:"<<li.size()<<endl;

while(!li.empty())

{

cout<<li.front()<<"\t";

li.pop\_front();

}

cout<<endl;

}

break;

case 3:

{

cout<<"deque operation\n";

cout<<"pop\_back(),push\_back(),back(),push\_front(),front(),pop\_front(),size(),empty stl list functions\n";

deque <int> de;

de.push\_back(30);

de.push\_back(40);

de.push\_back(50);

de.push\_front(20);

de.push\_front(10);

int q=de.size();

cout<<"Size of dequeue :"<<q<<endl;

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

{

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

cout<<endl;

}

de[2]=33;

while(!de.empty())

{

cout<<de.front()<<"\t";

de.pop\_front();

}

cout<<endl;

}

break;

case 4:

{

cout<<"more list operation\n";

cout<<"reverse(),merge(),unique() stl list functions\n";

int a[]={10,20,30,40,50};

int n=sizeof(a)/sizeof(a[0]);

list<int> li;

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

{

li.push\_front(a[i]);

}

cout<<"before merging \n";

cout<<"size of list li is:"<<li.size()<<endl;

list<int> l1;

l1.push\_front(35);

l1.push\_front(30);

l1.push\_front(25);

l1.push\_front(20);

l1.push\_front(15);

l1.reverse();

li.merge(l1);

li.unique();

while(!li.empty())

{

cout<<li.front()<<"\t";

li.pop\_front();

}

cout<<"\n";

cout<<"before merging \n";

cout<<"size of list li is:"<<li.size()<<endl;

}

break;

case 5:

{

exit(0);

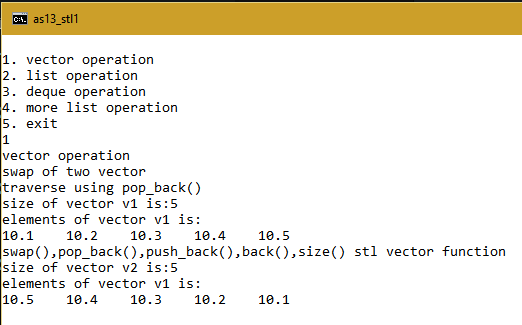
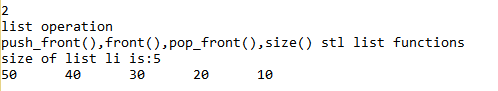
}

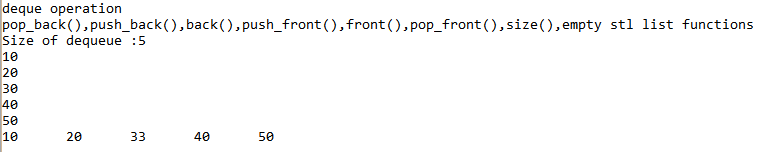
}

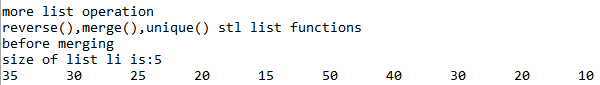
}

return 0;

}





Q-wap to show the functioning of standard library function(stl-2) list/vector/deque with iterator?

Ans-

#include<iostream>

#include<list>

#include<algorithm>

using namespace std;

int main()

{

int arr[]={1,2,3,4};

list<int> li;

int n=sizeof(arr)/sizeof(arr[0]);

cout<<"array of integers: is\n";

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

{

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

li.push\_back(arr[i]);

}

cout<<'\n';

cout<<"no. of elements in list\n"<<n<<endl;

cout<<"size oflist li:"<<sizeof(li)<<endl;

list<int>::iterator it;

cout<<"printing of list 1 using iterator \n";

for(it=li.begin();it!=li.end();it++)

{

cout<< \*it<<"\t";

}

cout<<"\n \n";

list<int> li1(4);

static int data=1;

cout<<"inserting data into list 2 using iterator\n";

for(it=li1.begin();it!=li1.end();it++)

{

data\*=2;

\*it=data;

}

cout<<"printing of list 2 using iterator\n";

for(it=li1.begin();it!=li1.end();it++)

{

cout<< \*it<<"\t";

}

cout<<"\n \n";

cout<<"special list iterator revesre function\n";

list<int>::reverse\_iterator ti;

cout<<"reversing the elements of list 1\n";

for(ti=li.rbegin();ti!=li.rend();ti++)

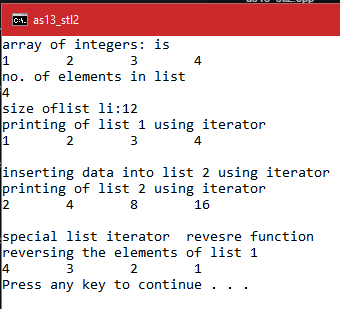
{

cout<<\*ti<<"\t";

}

return 0;

}



Q-wap to show the functioning of standard library function (stl\_ac) list with iterator?

#include<iostream>

#include<list>

#include<algorithm>

using namespace std;

class person

{

private:

string fname;

string lname;

int age;

public:

person(string l,string f,int a)

{

fname=f;

lname=l;

age=a;

}

void display()

{

cout<<"person name: "<<fname <<lname<<endl;

cout<<"person age is: "<<age<<endl;

}

void operator << (ostream &out)

{

out<<"neme of person"<<fname <<lname <<endl;

out<<"person age is:"<<age<<endl;

}

};

int main()

{

person p1("jhon","doe",18);

person p2("captain","clove",18);

list<person> li;

li.push\_back(p1);

li.push\_back(p2);

list<person>::iterator it;

for(it=li.begin();it!=li.end();it++)

{

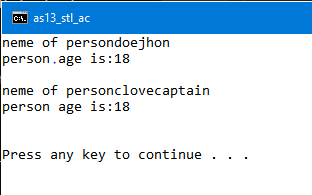
\*it<<cout;

cout<<"\n";

}

return 0;

}



Q-wap to show the functioning of standard library function (stl\_ac\_set) SET with iterator?

Ans-

#include<iostream>

#include<list>

#include<set>

#include<algorithm>

using namespace std;

int main()

{

string state[]={"punjab","gujrat","rajisthan","assam","chennai"};

int n=sizeof(state)/sizeof(state[0]);

set<string,less<string> >states;

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

{

states.insert(state[i]);

}

set<string,less<string> >::iterator it;

for(it=states.begin();it!=states.end();it++)

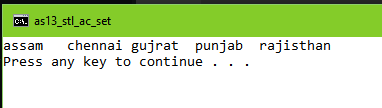
{

cout<<\*it<<"\t";

}

return 0;

}



Q-wap to show the functioning of standard library function (stl\_ac\_multiset) MULTI\_SET with iterator?

Ans-

#include<iostream>

#include<list>

#include<set>

#include<algorithm>

#include <iterator>

using namespace std;

class person

{

private:

string fname;

string lname;

int age;

public:

person(string l,string f,int a)

{

fname=f;

lname=l;

age=a;

}

bool operator < (const person &p) const

{

return(age<p.age);

}

ostream& operator << (ostream &out) const

{

out<<"name of the person"<<fname<<lname<<endl;

out<<"age of person"<<age<<endl;

return out;

}

void display() const

{

cout<<"neme of person: "<<fname<<" "<<lname <<endl;

cout<<"person age is: "<<age<<endl;

}

};

int main()

{

person p1("jhon","doe",18);

person p2("captain","marvel",20);

multiset <person, less <person> > mu;

mu.insert(p1);

mu.insert(p2);

multiset <person, less <person> > :: iterator it;

for(it=mu.begin(); it!=mu.end(); it++)

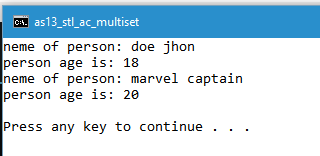
{

it->display();

}

return 0;

}



Q-wap to show the functioning of standard library function (stl\_ac\_map) MAP with iterator?

#include<iostream>

#include<list>

#include<set>

#include<algorithm>

#include <iterator>

using namespace std;

class person

{

private:

string fname;

string lname;

int age;

public:

person(string l,string f,int a)

{

fname=f;

lname=l;

age=a;

}

bool operator < (const person &p) const

{

return(age<p.age);

}

ostream& operator << (ostream &out) const

{

out<<"name of the person"<<fname<<lname<<endl;

out<<"age of person"<<age<<endl;

return out;

}

void display() const

{

cout<<"neme of person: "<<fname<<" "<<lname <<endl;

cout<<"person age is: "<<age<<endl;

}

};

int main()

{

person p1("jhon","doe",18);

person p2("captain","marvel",20);

multiset <person, less <person> > mu;

mu.insert(p1);

mu.insert(p2);

multiset <person, less <person> > :: iterator it;

for(it=mu.begin(); it!=mu.end(); it++)

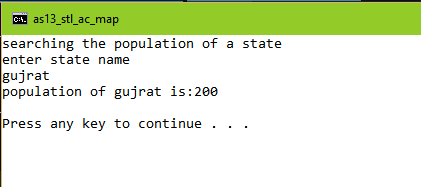
{

it->display();

}

return 0;

}



**Lab Assigment-14**

**Q-wap in c to show functionality of static data member &member function**

**ANS-**

#include<iostream>

using namespace std;

class point

{

private:

int x,y;

static int cnt;

public:

point()

{ ++cnt;

x=y=0;

}

point(int a, int b)

{

++cnt;

x=a;

y=b;

}

point(const point &p0)

{

++cnt;

x=p0.x;

y=p0.y;

}

static int get\_cnt()

{

return cnt;

}

};

int point::cnt=0;

int main()

{

point p1;

point p2(10,20);

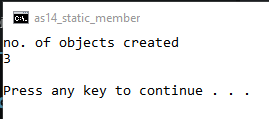
point p3(p2);

int c=point::get\_cnt();

cout<<"no. of objects created\n"<<c<<endl;

return 0;

}



**Q-wap in c to show the functionality of friend function in c++**

#include<iostream>

using namespace std;

class point

{

private:

int x,y;

public:

friend void func(point &po);

point(int a,int b)

{

cout<<"settting of points through parameterized constructor\n";

x=a;

y=b;

}

};

void func(point &po)

{

cout<<"accessing private data memmbers through friend function";

cout<<"x: "<<po.x<<endl;

cout << "y: "<<po.y<<endl;

}

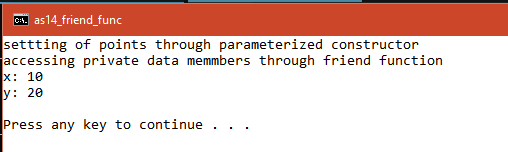
int main()

{

point p1(10,20);

func(p1);

}



**Q-wap in c to show file stream operation by writing and reading from a file**

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

fstream st;

char text[50];

st.open("file\_stream.txt",ios::out);

if(!st)

{

cout<<"File creation failed\n";

}

else

{

cout<<"Creation of file and Writing to file\n";

cout<<"write content to file\n";

cin.getline(text,50);

st<<text;

st.close();

}

st.open("file\_stream.txt",ios::in);

if(!st)

{

cout<<"No such file\n";

}

else

{

cout<<"Opening file and Reading from a file\n";

char ch;

while (!st.eof())

{

st >>ch;

cout << ch;

}

st.close();

}

return 0;

}

