**EVEREST ENGINEERING COLLEGE**

  **SANEPA, LALITPUR**

(AFFILIATED TO POKHARA UNIVERSITY)

AN

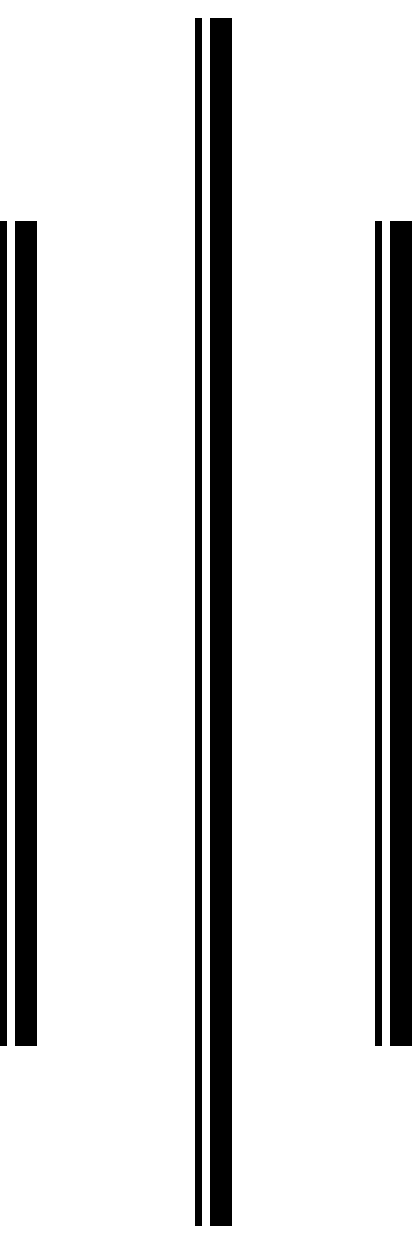
LAB REPORT

ON

**Object Oriented Programming In C++**

***[Dynamic Memory Allocation]***

**Lab Sheet: 6**



**SUBMITTED BY SUBMITTED TO**

Name:-**SAKCHYAM ACHARYA** Er.Pradip Paudel

Roll No:-40 Everest Engineering College

Department of Computer Engineering

Faculty:-BE CMP

Year/Semester: - 2nd SEM

Batch: 2021

Verified By:-

Student's Signature

**Index Page:-**

Topics Page No

* Title, Objective....................................................................................3
* Theory……………………………………………………………………………….3

• Dynamic memory allocation by new and delete operator

• Dynamic Constructor

• Dynamic initialization of Object

* Questions with Source Code & Output…………………………………………...3-10
* Discussion & Conclusion ...................................................................................11

**Title:** **Dynamic memory allocation.**

**Objective:**

• To be familiar with Dynamic memory allocation with C++.

• To be familiar with Dynamic constructor and Dynamic initialization of Object.

**Theory:**

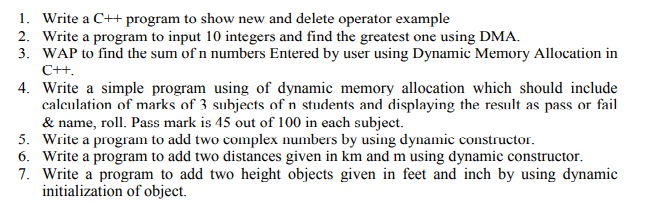
• **Dynamic memory allocation by new and delete operator:-** Dynamic memory allocation in C++ refers to performing memory allocation manually by a programmer. Dynamically allocated memory is allocated on Heap, and non-static and local variables get memory allocated on Stack.

New operator is use to allocate new memory whereas delete operator is use to deallocate dynamically allocated memory, programmers are provided delete operator in C++ language.

• **Dynamic Constructor:-** Dynamic constructor is used to allocate the memory to the objects at the run time. Memory is allocated at run time with the help of 'new' operator. By using this constructor, we can dynamically initialize the objects.

• **Dynamic initialization of Object:-** Dynamic initialization of object refers to initializing the objects at a run time i.e., the initial value of an object is provided during run time. It can be achieved by using constructors and by passing parameters to the constructors.

* **Lab Problems:**



**///Prob1)**

**//Source Code:**

#include<iostream>

using namespace std;

int main(){

int \*ptr=new int;

cout<<"Enter the value:";

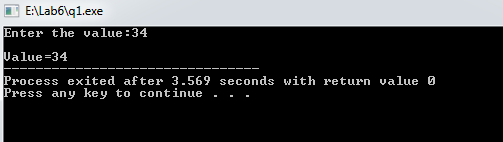
cin>>\*(ptr);

cout<<endl<<"Value="<<\*ptr;

delete ptr;

return 0;

}



**///Prob2)**

**//Source Code:**

#include<iostream>

using namespace std;

int main(){

int \*ptr=new int[10];

int gtr=0;

for (int i=0;i<10;i++)

{

cout<<"Enter the value of "<<i+1<<" number:";

cin>>ptr[i];

if (ptr[i]>gtr){

gtr=ptr[i];

}

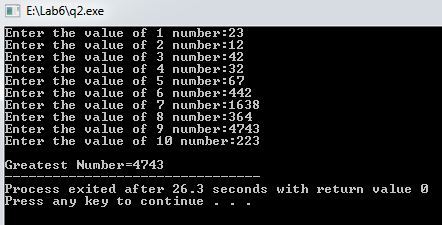
}

cout<<endl<<"Greatest Number="<<gtr;

delete []ptr;

return 0;

}



**///Prob3)**

**//Source Code:**

#include<iostream>

using namespace std;

int main(){

int \*ptr=new int[10],i=0,sum=0;

for (i=0;i<10;i++)

{

cout<<"Enter the value of "<<i+1<<" number:";

cin>>ptr[i];

sum=sum + ptr[i];

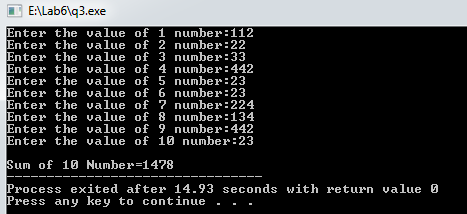
}

cout<<endl<<"Sum of "<<i+1<<" Number="<<sum;

delete []ptr;

return 0;

}



**///Prob4)**

**//Source Code:**

#include<iostream>

using namespace std;

class Student{

private:

char name[30];

short int rno;

float m1,m2,m3;

public:

void getData(void){

cin.ignore();

cout<<endl<<"Name?: ";

cin.get(name,30);

cout<<"Roll number?: ";

cin>>rno;

cout<<"Marks secured by the student in 3 different subjects: ";

cin>>m1>>m2>>m3;

}

void Display(void){

cout<<"\n\t\t\tStudent detail: \n";

cout<<"Student name:"<<name<<"\nStudent roll number:"<<rno;

if ((m1>45)&&(m2>45)&&(m3>45)){

cout<<"\nResult= Pass!!!\n";

}

else{

cout<<"\nResult status=Fail\*\*\*\n";

}

}

};

int main(){

cout<<"Enter the number of the student: "<<endl;

short int n;

cin>>n;

Student \*ptr=new Student[n];

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

ptr[i].getData();

}

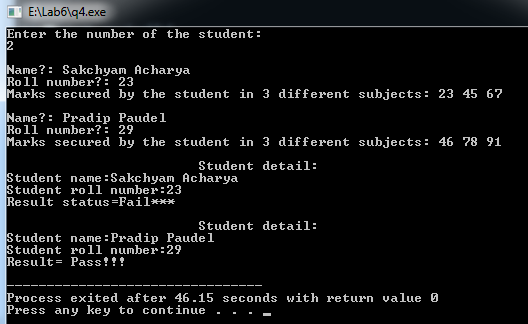
for (int i=0;i<n;i++){

ptr[i].Display();

}

return 0;

}



**///Prob5)**

**//Source Code:**

//WAP to add two complex number using dynamic constructor.

#include<iostream>

using namespace std;

class complex{

private:

int \*real,\*imag;

public:

complex(){

real=new int;

imag=new int;

\*real=0;

\*imag=0;

};

complex(int r,int i){

real=new int;

imag=new int;

\*real=r;

\*imag=i;

};

void addcomplex(complex c1,complex c2){

cout<<"First Complex Number:"<<\*c1.real<<"+"<<\*c1.imag<<"i"<<endl;

\*real=\*c1.real + \*c2.real;

cout<<"Second Complex Number:"<<\*c2.real<<"+"<<\*c2.imag<<"i"<<endl;

\*imag=\*c1.imag + \*c2.imag;

delete c1.real,c1.imag,c2.real,c2.imag;

};

void display(){

cout<<endl<<"\t\tSum of two complex number:"<<\*real<<"+"<<\*imag<<"i"<<endl;

delete real,imag;

};

};

int main(){

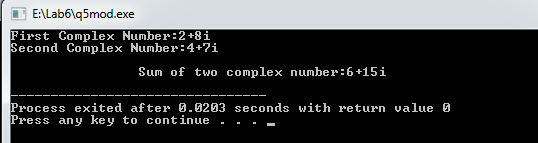
complex c1(2,8), c2(4,7), c3;

c3.addcomplex(c1,c2);

c3.display();

return 0;

}



**///Prob6)**

**//Source Code:**

//WAP to add two distance given in km & m using dynamic constructor.

#include<iostream>

using namespace std;

class Distance{

private:

int km,m;

public:

Distance(){

int \*ptr=new int[2];

static int i=0;

i++;

cout<<endl<<"Enter the "<<i<<" Distance number:";

cout<<endl<<"KiloMeter?:";

cin>>ptr[0];

km=ptr[0];

cout<<"Meter?:";

cin>>ptr[1];

m=ptr[1];

delete[]ptr;

}

void adddis(Distance d1,Distance d2){

m=d1.m + d2.m;

km=d1.km + d2.km;

km=km+(m/1000);

m%=1000;

}

void display(){

cout<<endl<<"\t\tSum of Distance:";

cout<<endl<<"KiloMeter:"<<km<<endl<<"Meter:"<<m<<endl;

}

};

int main(){

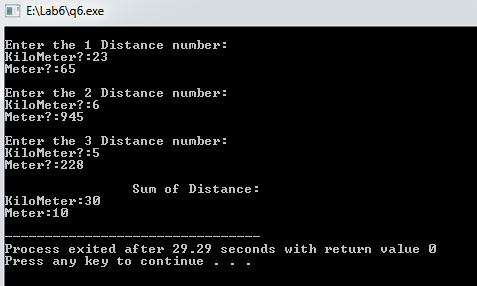
Distance d1,d2,d3;

d3.adddis(d1,d2);

d3.display();

return 0;

}



**///Prob7)**

**//Source Code:**

//WAP to add two height given in ft & inch using dynamic constructor.

#include<iostream>

using namespace std;

class height{

private:

int ft,in;

public:

height(){

int \*ptr=new int[2];

static int i=0;

i++;

cout<<endl<<"Enter the "<<i<<" Height Details:";

cout<<endl<<"Foot?:";

cin>>ptr[0];

ft=ptr[0];

cout<<"Inch?:";

cin>>ptr[1];

in=ptr[1];

delete [] ptr;

}

void addhei(height h1,height h2){

in=h1.in + h2.in;

ft=h1.ft + h2.ft;

ft+=(in/12);

in%=12;

}

void display(){

cout<<endl<<"\t\tSum of Height:";

cout<<endl<<"FOOT:"<<ft<<endl<<"Inch:"<<in<<endl;

}

};

int main(){

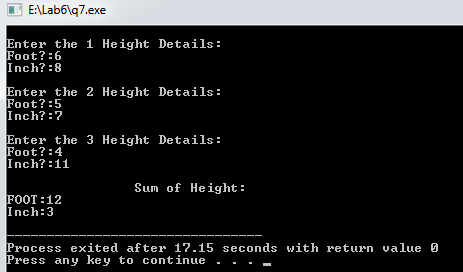
height h1,h2,h3;

h3.addhei(h1,h2);

h3.display();

return 0;

}



**Discussion & Conclusion:-**

The program is focused on various tasks on “**Dynamic Memory Allocation**”. From this program I understood how to allocate memory dynamically by new & delete operator & I have also learnt about dynamic constructor, dynamic initiliaziling of object.