# Experiment No.11 Operator Overloading

## Program 12

**Aim**: Create a c++ program that has a class product (prod\_id, name, price), which is inherited by other classes viz., belt ( length, color, width), shoes ( size, formal/casual, laced/non laced) and caps (diameter, water proof/not, color).List out the products available to the user and allow the user to select the products and the quantity. Overload\* operator for multiplying quantity with objects. Display the product wise price and total amount.

### Program:

#include <string.h> #include <iostream> using namespace std; class product

{

int q, size;

char brand[20]; float price, tprice;

public:

int prod\_id; product()

{

q = prod\_id = size = 0;

strcpy(brand, "\0"); price = tprice = 0.0;

}

void operator\*(int n)

{

q = n;

tprice = price \* n;

}

float operator+(float grand)

{

grand = grand + tprice; return grand;

}

void acc(); void print();

} \* p[3];

void product::acc()

{

cout << "\nEnterProduct\_Id:"; cin >> prod\_id;

cout << "\nEnterBrand:";

fflush(stdin); gets(brand);

cout << "\nEnterSize:"; cin >> size;

cout << "\nEnterPrice:"; cin >> price;

}

void product::print()

{

cout << "\nProductId:" << prod\_id << "\t\tBrand:" << brand << "\t\tSize:" << size << "\tPrice:" << price;

if (q!= 0)

cout << "\tQUANTITY:" << q;

}

class shoes : public product

{

public:

string type,lace; shoes()

{

cout << "\nEnter Details For Shoe:\n";

acc();

cout<<"\nEnter type of shoe(Formal/Casual):"; cin>>type;

cout<<"\nEnter type of shoe(Laced/Non-Laced):"; cin>>lace;

}

int returnid()

{

return prod\_id;

}

void display()

{

cout << "\nDISPLAYING SHOE DETAILS:-\n";

print();

cout<<"\tShoe Type:"<<type<<" "<<lace;

}

};

class belts : public product

{

public:

int len,width;

string color; belts()

{

cout << "\nEnter Details For Belt:\n"; acc();

cout<<"\nEnter the length:"; cin>>len;

cout<<"\nEnter the color:"; cin>>color;

cout<<"\nEnter the width:"; cin>>width;

}

int returnid()

{ return prod\_id;

}

void display()

{

cout << "\nDISPLAYING BELT DETAILS:-\n";

print(); cout<<"\tLength:"<<len<<"\t"; cout<<"Color:"<<color<<"\t";

cout<<"Width:"<<width<<"\t";

}

};

class caps : public product

{

public:

float diameter; string wp,color; caps()

{

cout << "\nEnter Details For Cap:\n"; acc();

cout<<"\nEnter Diameter:"; cin>>diameter;

cout<<"\nEnter if waterproof:"; cin>>wp;

cout<<"\nEnter Color:"; cin>>color;

}

int returnid()

{

return prod\_id;

}

void display()

{

cout << "\nDISPLAYING CAP DETAILS:\n";

print(); cout<<"\tDiameter:"<<diameter<<"\t"; cout<<"Waterproof:"<<wp<<"\t"; cout<<"Color:"<<color;

}

};

int main()

{

int i, id, n;

float grand = 0.0; shoes s[1];

belts b[1]; caps c[1];

cout << "\nOptionsForBuying"; cout << "\n\n------SHOES ";

for (i = 0; i < 1; i++)

s[i].display();

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

for (i = 0; i < 1; i++) b[i].display();

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

for (i = 0; i < 1; i++) c[i].display();

cout << "\nSelect by putting id and quantity for shoes:\n"; cin >> id >> n;

for (i = 0; i < 1; i++)

if (s[i].returnid() == id)

{

p[0] = &s[i];

\*p[0] \* n; break;

}

else

{

p[0] = &s[0];

\*p[0] \* 0;

}

cout << "\nSelect by putting id and quantity for belts:\n"; cin >> id >> n;

for (i = 0; i < 1; i++)

if (b[i].returnid() == id)

{

p[1] = &b[i];

\*p[1] \* n; break;

}

else

{

p[1] = &b[0];

\*p[1] \* 0;

}

cout << "\nSelect by putting id and quantity for caps:"; cin >> id >> n;

for (i = 0; i < 1; i++)

if (c[i].returnid() == id)

{

p[2] = &c[i];

\*p[2] \* n;

break;

}

else

{

p[2] = &c[0];

\*p[2] \* 0;

}

for (i = 0; i < 3; i++)

{

p[i]->print();

grand = \*p[i] + grand;

}

cout << "\n\n\tGRAND TOTAL:" << grand; return 0;

}

**Sample Input Output:** Enter Details For Shoe: EnterProduct\_Id: 34 EnterBrand: Puma EnterSize: 6

EnterPrice: 2500

Enter type of shoe(Formal/Casual): Casual Enter type of shoe(Laced/Non-Laced): Laced Enter Details For Belt:

EnterProduct\_Id: 11 EnterBrand: Levis EnterSize: 30

EnterPrice: 1200 Enter the length: 32 Enter the color: Brown Enter the width: 4 Enter Details For Cap: EnterProduct\_Id: 65 EnterBrand: Nike EnterSize: 7

EnterPrice: 999

Enter Diameter: 15 Enter if waterproof: yes Enter Color: Red OptionsForBuying

SHOES

DISPLAYING SHOE DETAILS:-

ProductId:34 Brand: Puma Size:06 Price:2500 Shoe Type:Casual Laced

BELTS

DISPLAYING BELT DETAILS:-

ProductId:11 Brand: Levis Size:30 Price:1200 Length:32 Color:Brown Width:4

CAPS

DISPLAYING CAP DETAILS:

ProductId:65 Brand: Nike Size:07 Price:999 Diameter:15 Waterproof: yes Color:Red

Select by putting id and quantity for shoes: 34

10

Select by putting id and quantity for belts: 11

10

Select by putting id and quantity for caps:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 65 |  | | | |
| 10 |
| ProductId:34 | Brand: Puma | Size:06 | Price:2500 | QUANTITY:10 |
| ProductId:11 | Brand: Levis | Size:30 | Price:1200 | QUANTITY:10 |
| ProductId:65 | Brand: Nike | Size:07 | Price:999 | QUANTITY:10 |

GRAND TOTAL:46990

## Program 13

**Aim:** Write a c++ program that maintains the date in (day,mon,year)format and overload the

++&--operator to view previous or next date from today’s date. Ensure that the day won’t exceed 30/31 for certain months.

### Program:

#include<iostream> #include<conio.h> using namespace std; class date

{

int day; int month; int year; bool flag; public:

date()

{

cout<<"enter the day,month and year in DD/MM/YYYY format:-"; cin>>day>>month>>year;

}

void operator ++()

{

flag=true;

if(day==29 && month==2)

{

month =3; day=1; show();

}

if(day==28 &&month ==2)

{

if(year%4==0)

{

day++; show();

}

else

{ month =3; day=1; show();

}

}

if(day==30 &&(month==4||month==6||month==9||month==11))

{ day=1;month++;show();

}

if(day==31&&month==12)

{

day=1; year++; month=1; show();

}

else if(day==31)

{ month++; day=1; show();

}

if(flag)

{ day++; show();

}

};

void operator --()

{

flag=true;

if(day==1 && month==3 && year%4==0)

{

day=29; month--;

show();

}

if(day==1 && month ==3)

{

day=28; month--;

show();

}

if(day==1 &&(month==5||month==7||month==10||month==12))

{

day=30; month--;

show();

}

if(day==1&&month==1)

{

year--; day=31; month=12; show();

}

if(day==1)

{

month--; day=31; show();

}

if(flag)

{

day--;

show();

}

};

void show()

{

flag=false;

cout<<"DAY :"<<day<<endl<<"MONTH :"<<month<<endl<<"YEAR :"<<year<<endl;

}

}date;

int main()

{ int ch;

while(1)

{

cout<<"\n1.INCREMENT\n2.DECREMENT\n3.EXIT\nEnter Choice :"; cin>>ch;

switch(ch)

{

case 1:++date; break;

case 2:--date; break;

case 3:return 0; break;

default: cout<<"\nWrong Input !!!";

}

}

}

### Sample Input Output:

enter the day,month and year in DD/MM/YYYY format:-26 12 2021 1.INCREMENT

1. DECREMENT
2. EXIT

Enter Choice :1

DAY :27

MONTH :12

YEAR :2021

1. INCREMENT
2. DECREMENT
3. EXIT

Enter Choice :2

DAY :26

MONTH :12

YEAR :2021

1. INCREMENT
2. DECREMENT
3. EXIT

Enter Choice :3

# Experiment No.12

**Virtual Function ,Virtual Base Class And Function Overloading**

## Program 14

**Aim:** Create a class powered device (type (electronic/mechanical), power range (10W- 20W), self\_powered/adapter) and inherit it in two classes scanner (no. of pages per second, resolution, size, cost, brand) and printer (brand, dpi, color/b/w, pages per second, laser/inkjet). Create another class named copier that inherits the features of printer and scanner along with additional features (storage capacity, stored copy printing speed). Ensure that the duplicate copies of powered\_device object don’t exist in copier.

### Program:

#include <iostream> using namespace std; class powerdev

{

char type[15], range[10], typepow[20];

public:

bool flag1, flag2; powerdev()

{

flag1 = false; flag2 = false;

}

void acc()

{

if (!flag1)

{

cout << "\n\nEnter Details of power device"; cout << "\nEnterType(electronic/mechanical):"; fflush(stdin);

gets(type);

cout << "Enter power range(15w-20w):"; fflush(stdin);

gets(range);

cout << "Enter Power Type(self-powered/adapter):"; fflush(stdin);

gets(typepow); flag1 = true;

}

}

void dis()

{

if (!flag2)

{

cout << "\n\nDetails of power device"; cout << "\nType:";

fflush(stdout); puts(type);

cout << "\nPower range(15w-20w):"; fflush(stdout);

puts(range);

cout << "\nPower Type(self-powered/adapter):"; fflush(stdout);

puts(typepow); flag2 = true;

}

}

};

class scanner : public virtual powerdev

{

int nop, resol1, resol2, size, cost; char brand[20];

public:

void accept1()

{

acc();

cout << "\n\n-----Enter Details of Scanner ";

cout << "\nEnter No Of Pages/second:"; cin >> nop;

cout << "\nEnter Resolution1:"; cin >> resol1;

cout << "\nEnter Resolution2:"; cin >> resol2;

cout << "\nEnter Size:"; cin >> size;

cout << "\nEnter Brand:"; fflush(stdin); gets(brand);

}

void disp1()

{

dis();

cout << "\n\n-----Details of Scanner ";

cout << "\nNo Of Pages/second:" << nop;

cout << "\nResolution:" << resol1 << "x" << resol2; cout << "\nSize:" << size;

cout << "\nBrand:" << brand;

}

};

class printer : public virtual powerdev

{

int nop, dpi;

char brand[20], color[10], type[10];

public:

void accept2()

{

acc();

cout << "\n\nEnter Details of Printer"; cout << "\nEnter No Of Pages/second:"; cin >> nop;

cout << "\nEnter DPI:"; cin >> dpi;

cout << "\nEnter Brand:"; fflush(stdin); gets(brand);

cout << "Enter Color(coloured/black-white):"; fflush(stdin);

gets(color);

cout << "Enter Type(laser/inkjet):"; fflush(stdin);

gets(type);

}

void disp2()

{

dis();

cout << "\n\nDetails of Printer";

cout << "\nNo Of Pages/second:" << nop; cout << "\nDPI:" << dpi;

cout << "\nColor:" << color; cout << "\nType:" << type; cout << "\nBrand:" << brand;

}

};

class copier : public scanner, public printer

{

int store, speed;

public:

void enter()

{

cout << "\nEnter Copier Details"; accept1();

accept2();

cout << "\nEnter Speed/second:"; cin >> speed;

cout << "\nEnter Storage Capacity:"; cin >> store;

}

void display()

{

cout << "\nCopier Details:"; disp1();

disp2();

cout << "\nSpeed/persecond:" << speed; cout << "\nStorageCapacity:" << store;

}

};

int main()

{

copier c; c.enter();

c.display(); return 0;

}

### Sample Input Output:

Enter Copier Details

Enter Details of power device

EnterType(electronic/mechanical): Electronic Enter power range(15w-20w):20

Enter Power Type(self-powered/adapter): adapter

-----Enter Details of Scanner-----

Enter No Of Pages/second:20 Enter Resolution1:200

Enter Resolution2:300 Enter Size:10

Enter Brand:cannon

Enter Details of Printer

Enter No Of Pages/second:10 Enter DPI: 100

Enter Brand:cannon

Enter Color(coloured/black-white): black-white Enter Type(laser/inkjet): laser

Enter Speed/second:15 Enter Storage Capacity:512

Copier Details:

Details of power device Type: Electronic

Power range(15w-20w):20

Power Type(self-powered/adapter): adapter

-----Details of Scanner-----

No Of Pages/second:20 Resolution:200x300 Size:10

Brand:cannon

Details of Printer

No Of Pages/second:10 DPI:100

Color: black-whi laser Type: laser Brand:cannon Speed/persecond:15 StorageCapacity:512

## Program 15

**Aim:** Create a class telephone\_directory (name, occupation, phone, address) and an virtual function search (), that searches the object contents in a linear fashion. Extend the class in a specific class called doctor (qualification, clinic, visiting time) and override the virtual function such that it searches using binary search. Also create another class lawyer (qualification, civil/criminal, cases attended, contact\_no, office\_address) that extends the telephone\_directory such that it won’t override the virtual function. Search for a doctor and a lawyer by taking the option from the user.

### Program:

#include <bits/stdc++.h> using namespace std;

int dn, ln;

class telephone\_directory

{

public:

char name[30], occ[10], add[50]; long long int phone;

virtual void search() = 0;

};

class doctor : public telephone\_directory

{

public:

char quali[10], clinic[30], visit[20]; void accept()

{

cout << "\nEnter Name : "; fflush(stdin);

gets(name);

cout << "Enter Occupation : "; fflush(stdin);

gets(occ);

cout << "Enter Phone No : "; cin >> phone;

cout << "Enter Address : "; fflush(stdin);

gets(add);

cout << "Enter Qualification : "; fflush(stdin);

gets(quali);

cout << "Enter Clinic : "; fflush(stdin); gets(clinic);

cout << "Enter Visiting Hours : "; fflush(stdin);

gets(visit);

}

void display()

{

cout << "\nName : " << name; cout << "\nOccupation : " << occ; cout << "\nPhone No : " << phone; cout << "\nAddress : " << add;

cout << "\nQualification : " << quali; cout << "\nClinic : " << clinic;

cout << "\nVisiting Hours : " << visit;

}

void search();

} d[50];

void doctor::search()

{

int i;

char n[30];

cout << "\nEnter Name Of Doctor To Be Searched : "; fflush(stdin);

gets(n);

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

if (strcmp(d[i].name, n) == 0)

{

cout << "\nFound\n"; d[i].display();

break;

}

if (i == dn)

cout << "\nNotFound\n";

}

class lawyer : public telephone\_directory

{

public:

char quali[15], typ[10], offadd[50];

int casatten; long int contac; void accept()

{

cout << "\nEnter Name : "; fflush(stdin);

gets(name);

cout << "Enter Occupation : "; fflush(stdin);

gets(occ);

cout << "Enter Phone No : "; cin >> phone;

cout << "Enter Address : "; fflush(stdin);

gets(add);

cout << "Enter Qualification : "; fflush(stdin);

gets(quali);

cout << "Enter Type(civil/criminal) : "; fflush(stdin);

gets(typ);

cout << "Enter Cases attended : "; cin >> casatten;

cout << "Enter Contactno : "; cin >> contac;

cout << "Enter Office Address : "; fflush(stdin);

gets(offadd);

}

void display()

{

cout << "\nName : " << name; cout << "\nOccupation : " << occ; cout << "\nPhoneNo : " << phone; cout << "\nAddress : " << add;

cout << "\nQualification : " << quali; cout << "\nType : " << typ;

cout << "\nCases Attended : " << casatten; cout << "\nContact No : " << contac;

cout << "\nOffice Address : " << offadd;

}

void search();

} l[50], temp;

void lawyer::search()

{

bool flag = false; int mid, low, high; char n[30];

cout << "\nEnter Name Of lawyer To Be Searched : "; fflush(stdin);

gets(n);

for (low = 0; low < ln; low++)

for (mid = 0; mid < ln - low - 1; mid++)

if (strcmp(l[mid].name, l[mid + 1].name) > 0)

{

temp = l[mid]; l[mid] = l[mid + 1]; l[mid + 1] = temp;

}

low = 0; high = ln;

while (low < high)

{

mid = (low + high) / 2;

if (strcmp(l[mid].name, n) == 0)

{

cout << "\nFound\n"; l[mid].display();

flag = true; break;

}

else if (strcmp(l[mid].name, n) > 0)

{

high = mid - 1;

}

else if (strcmp(l[mid].name, n) < 0)

{

low = mid + 1;

}

}

if (!flag)

cout << "\nNot Found\n";

}

int main()

{

int i;

cout << "\nEnter No Of Doctors : "; cin >> dn;

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

{

cout << "\nEnter Details For Doctor" << i + 1 << ":"; d[i].accept();

}

cout << "\nEnter No Of Lawyers : "; cin >> ln;

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

{

cout << "\nEnter Details For Lawyer" << i + 1 << ":"; l[i].accept();

}

cout << "\nWhat Do You Want To Search For 1.Doctor 2.Lawyer\nEnter Your Choice : "; cin >> i;

if (i == 1) d[0].search();

else if (i == 2)

l[0]

.search(); return 0;

}

### Sample Input Output:

Enter No Of Doctors : 2

Enter Details For Doctor1: Enter Name : XYZ

Enter Occupation : Doctor Enter Phone No : 8094xxxxxx Enter Address :Udaipur

Enter Qualification : MBBS Enter Clinic : JKL

Enter Visiting Hours : 8 to 12

Enter Details For Doctor2: Enter Name : ABC

Enter Occupation : Doctor Enter Phone No : 9410xxxxxx Enter Address : Udaipur

Enter Qualification : BDS Enter Clinic : ABC

Enter Visiting Hours : 12 to 4

Enter No Of Lawyers : 2

Enter Details For Lawyer1:

Enter Name : HIJ

Enter Occupation : Lawyer Enter Phone No : 9438xxxxxx Enter Address : Udaipur

Enter Qualification : LLB

Enter Type(civil/criminal) : Civil Enter Cases attended : 18

Enter Contactno : 9438xxxxxx Enter Office Address : Jaipur

Enter Details For Lawyer2:

Enter Name : TUV

Enter Occupation : Lawyer Enter Phone No : 9358xxxxxx Enter Address : Dabok

Enter Qualification : LLB

Enter Type(civil/criminal) : Criminal Enter Cases attended : 100

Enter Contactno : 9358xxxxxx Enter Office Address : Dabok

What Do You Want To Search For 1.Doctor 2.Lawyer Enter Your Choice : 1

Enter Name Of Doctor To Be Searched : XYZ

Found

Name : XYZ Occupation : Doctor Phone No : 8094xxxxxx Address : bhilwara Qualification : MBBS Clinic : JKL

Visiting Hours : 8 to 12

# Experiment No. 13

**Constant Data Member and Member Functions**

## Program 16

**Aim:** Create a class that holds the details of the hostel students like name, age, internal room phone no.(in 4digits) etc. Use a function of that class to check whether the mobile number isa prime and set the flag field of the class as 1. Finally display the objects that has their mobilenumber as prime. This function should be constant.

### Program:

#include <iostream> using namespace std; #define max 5

class student

{

int age;

char name[20]; int mob;

public:

int flag; student()

{

fflush(stdin);

cout << "\nENTER THE NAME OF THE STUDENT : ";

gets(name);

cout << "\nENTER THE AGE : ";

cin >> age;

cout << "\nENTER THE MOBILE NUMBER (first 4 digit) : "; cin >> mob;

flag = 1; checkprime();

}

void checkprime(); void display() const;

int returnflag()

{

return flag;

}

};

void student::display() const

{

fflush(stdout);

cout << "\nNAME OF THE STUDENT : ";

puts(name);

cout << "AGE : "; cout << age;

cout << "\nMOBILE NUMBER : ";

cout << mob;

}

void student::checkprime()

{

int i;

for (i = 2; i < mob / 2; i++)

{

if (mob % i == 0)

{

flag = 0; break;

}

}

}

int main()

{

int i;

const student s[max];

cout << "\nDETAILS OF STUDENTS WITH PRIME MOBILE NUMBERS : ";

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

{

if (s[i].flag == 1)

s[i].display();

}

return 0;

}

### Sample Input Output:

ENTER THE NAME OF THE STUDENT : XYZ ENTER THE AGE : 18

ENTER THE MOBILE NUMBER (first 4 digit) : 8094 ENTER THE NAME OF THE STUDENT :ABC ENTER THE AGE : 20

ENTER THE MOBILE NUMBER (first 4 digit) : 9437

ENTER THE NAME OF THE STUDENT : HIJ ENTER THE AGE : 19

ENTER THE MOBILE NUMBER (first 4 digit) : 9237 ENTER THE NAME OF THE STUDENT : DEF ENTER THE AGE : 20

ENTER THE MOBILE NUMBER (first 4 digit) : 1237 ENTER THE NAME OF THE STUDENT : KLM ENTER THE AGE : 20

ENTER THE MOBILE NUMBER (first 4 digit) : 8897

DETAILS OF STUDENTS WITH PRIME MOBILE NUMBERS : NAME OF THE STUDENT : ABC

AGE : 20

MOBILE NUMBER : 9437 NAME OF THE STUDENT : DEF AGE : 20

MOBILE NUMBER : 1237

# Experiment No.14

**Static Data Member and Member Functions**

## Program 17

**Aim:** Create a class ebill with the elements:-custno,pre\_read,cur\_read,total\_units,&amt . Read the required data for 5 customer and compute the total reading and amount to be paid by the 5 customer and print the result. (use the table to calculate total amount). Also display the customer’s details who have paid the highest bill using static data member.

### Consumption

**Units**

### Rate of

**Charge**

0-200 Rs. 0.50per unit

201-400 Rs.100 plus

401-600 Rs.230 plus

Rs 0.65per unit excess of 200

Rs 0.80 per

unit excess of 400

601 and above Rs. 390 plus Rs 1.00 per unit excess of 600

### Program:

#include <iostream> using namespace std; static float high = 0; class ebill

{

int custno, current, pre, total; float amt;

public:

void getdata()

{

cout << "Customer no."; cin >> custno;

cout << "Pre read:";

cin >> pre;

cout << "Current read:"; cin >> current;

}

void cal()

{

total = current - pre;

if (total >= 0 && total <= 200)

{

amt = total \* 0.5;

}

else if(total >= 201 && total <= 400)

{

amt = 100.00 + 0.65 \* (float)total;

}

else if(total >= 401 && total <= 600)

{

amt = 230.00 + 0.8 \* (float)total;

}

else if(total >= 601)

{

amt = 390.00 + 1.0 \* (float)total;

}

if (amt > high)

{

high = amt;

}

}

void showdata()

{

cout << "Customer no.:" << custno; cout << "\nTotal units:" << total;

cout << "\nAmount:" << amt << endl;

}

void highest()

{

if (amt == high)

{

cout << "\nCustomer no.:" << custno; cout << "\nTotal units:" << total; cout << "\nAmount:" << amt;

}

}

};

int main()

{

int i;

ebill e[5];

cout << "Enter customer details:\n"; for (i = 0; i < 5; i++)

{

cout << i + 1 << "."; e[i].getdata();

}

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

{

e[i].cal();

}

cout << "\nCustomer's database:\n";

{

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

{

cout << i + 1 << "."; e[i].showdata();

}

}

cout << "\n\nHighest paying customer:"; for (i = 0; i < 5; i++)

{

e[i].highest();

}

}

### Sample Input Output:

Enter customer details: 1.Customer no.1456 Pre read:1122

Current read:2233 2.Customer no.1567 Pre read:3344 Current read:5566 3.Customer no.3698 Pre read:1918 Current read:1915 4.Customer no.1846 Pre read:1789 Current read:9987

5.Customer no.1900 Pre read:7788 Current read:8899

Customer's database:

1. Customer no.:1456Total units:1111 Amount:1501 2.Customer no.:1567 Total units:2222 Amount:2612 3.Customer no.:3698 Total units:-3 Amount:5.96079 4.Customer no.:1846 Total units:8198 Amount:9587 5.Customer no.:1900 Total units:1313 Amount:1501

Highest paying customer: Customer no.:1846

Total units:8198 Amount:958

# Experiment No. 15 Templates

## Program 18

**Aim:** Create a class template for list that can store names or numbers of students. Also performthe operations like insert, delete, update and display by either overloading template functionsor using generic template functions.

### Program:

#include<iostream> #include<string.h> using namespace std;

template <typename T> class student{

private:

T stud[4]; public:

void insert(T val,int i){ stud[i]=val;

}

T deletion(int i){ T temp;

temp = stud[i]; return temp;

}

void update(T val,int i){ stud[i]=val;

}

void display(){

for(int i=0;i<3;i++){ cout<<stud[i]<<"\t";

}

cout<<endl;

}

};

int main(){

student <int> s1; student <string> s2; s1.insert(1234,0); s1.insert(4321,1); s1.insert(1111,2); s1.insert(1122,3); s1.deletion(3); s1.update(2315,3); s1.display(); s2.insert("XYZ",0);

s2.insert("ABC",1);

s2.insert("TUV",2);

s2.insert("FGH",3); s2.deletion(3); s2.update("IJK",3); s2.display();

}

### Sample Input Output:

1234 4321 1111

XYZ ABC TUV

# Experiment No. 16 Exception Handling

## Program 19

**AIM:** Create a class that maintains the details like, given\_no, a temporary array, and factorial of the given no. Use a function that finds the factorial of the given number without using recursion by making use of the temporary array. Handle the array overflow exception when all the locations of the array were used and display the message “internal memory is not sufficient to carry out this operation”.

### Program:

#include <iostream> using namespace std; class sample

{

int given\_no, factorial; int temp[5];

public:

void fact(int val)

{

given\_no = val; temp[0] = 1;

for (int i = 1; i <= given\_no; i++)

{

try

{

if(i>5){

throw i;

}

temp[i] = temp[i - 1] \* i; cout << temp[i] << "\t";

}

catch(int x){

cout<<"Internal memory is not sufficient to carry out this operation"<<endl; break;

}

}

if(given\_no<5)

display();

}

void display(){

factorial = temp[given\_no];

cout << "Factorial of " << given\_no << " is " << factorial << endl;

}

};

int main()

{

sample s; s.fact(1);

s.fact(2);

s.fact(3);

s.fact(4);

s.fact(5);

s.fact(6);

}

### Sample Input Output:

1 Factorial of 1 is 1

1 2 Factorial of 2 is 2

1 2 6 Factorial of 3 is 6

1 2 6 24 Factorial of 4 is 24

1 2 6 24 120 1 2 6 24 120 Internal memory is not sufficient to carry out this operation

# Experiment No. 17 File Handling

## Program 20

**Aim:** Write a C++ program using File Handling to perform the following operations:-

1. A file named “data.txt” contains numbers from 0-100. Open the “data.txt” file in read mode.
2. Read data from “data.txt” file.
3. If the number is odd, open a new file named “odd.txt” and write the odd number into “odd.txt” file.
4. If the number is even, open a new file named “even.txt” and write the odd number into “odd.txt” file.
5. Display the data of all the files.

**Program:** #include<iostream> #include<fstream> using namespace std; int main(){

fstream data,even,odd; int c,i;

data.open("data.txt",ios::out); for(int i=0;i<=100;i++){

data<<i<<"\t";

}

data.close(); data.open("data.txt",ios::in);

even.open("even.txt",ios::out);

odd.open("odd.txt",ios::out); while (!data.eof())

{

data>>c; if(c%2==0){

even<<c<<"\t";

}

else{

odd<<c<<"\t";

}

}

odd.close();

even.close(); odd.open("odd.txt",ios::in); char ch;

while(!odd.eof())

{

odd.get(ch); cout<<ch;

}

odd.close();

even.open("even.txt",ios::in);

while(!even.eof())

{

even.get(ch); cout<<ch;

}

return 0;

}

### Sample Input Output:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 | 21 | 23 | 25 | 27 | 29 | |
| 31 | 33 | 35 | 37 | 39 | 41 | 43 | 45 | 47 | 49 | 51 | | 53 | 55 | 57 | 59 |
| 61 | 63 | 65 | 67 | 69 | 71 | 73 | 75 | 77 | 79 | 81 | | 83 | 85 | 87 | 89 |
| 91 | 93 | 95 | 97 | 99 |  | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 |  |
| 18 | 20 | 22 | 24 | 26 | 28 | 30 | 32 | 34 | 36 | 38 | | 40 | 42 | 44 | 46 |
| 48 | 50 | 52 | 54 | 56 | 58 | 60 | 62 | 64 | 66 | 68 | | 70 | 72 | 74 | 76 |
| 78 | 80 | 82 | 84 | 86 | 88 | 90 | 92 | 94 | 96 | 98 | | 100 | 100 |  |  |

**RUBRICS EVALUATION**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Performance Criteria** | **Scale 1**  **(0-25%)** | **Scale 2**  **(26-50%)** | **Scale 3**  **(51-75%)** | **Scale 4**  **(76-100%)** | **Score (Numerical)** |
| **Understandability**  Ability to analyse Problem and Identify solution | Unable to understand the problem. | Able to understand the problem partially and unable to identify the solution | Able to understand the problem completely but unable to identify the solution | Able to understand the problem completely and able to provide alternative  solution too. |  |
| **Logic**  Ability to specify Conditions & control flow that are appropriate for the problem domain. | Program logic is incorrect | Program logic is on the right track but has several errors | Program logic is mostly correct, but may contain an occasional boundary error or redundant or  contradictory condition. | Program logic is correct, with no known boundary errors, and no redundant or contradictory conditions. |  |
| **Debugging**  Ability to execute  /debug | Unable to execute program | Unable to debug several errors. | Able to execute program with  several warnings. | Able to execute program completely |  |
| **Correctness** Ability to code formulae and algorithms that reliably produce correct answers or appropriate results. | Program does not produce correct answers or appropriate results for most inputs. | Program approaches correct answers or appropriate results for most inputs, but can contain miscalculations in some cases. | Program produces correct answers or appropriate results for most inputs. | Program produces correct answers or appropriate results for all inputs tested. |  |
| **Completeness** Ability to demonstrate and deliver on time. | Unable to explain the code.and the code was overdue. | Unable to explain the code and the code submission was late. | Able to explain code and the program was delivered within the due  date. | Able to explain code and the program was delivered on time. |  |
| **TOTAL** | | | | |  |

### Computer Lab’s Do’s and Don’ts and Safety Rules

**DO’s**

* Please switch off the Mobile/Cell phone before entering Lab.
* Check whether all peripheral are available at your desktop before proceeding for the session
* Arrange all the peripheral and seats before leaving the lab.
* Properly shutdown the system before leaving the lab.
* Keep the bag outside in the racks.
* Enter the lab on time and leave at proper time.
* Maintain the decorum of the lab.

### DON’TS

* Don’t mishandle the system.
* Don’t leave the system on standing for long
* Don’t bring any external material in the lab.
* Don’t make noise in the lab.
* Don’t bring the mobile in the lab.
* Don’t enter in the lab without permission of lecturer/laboratory technician immediately
* Don’t delete or make any modification in system files.
* Don’t bring storage devices like pen drive without permission of lecturer/laboratory technician.

### Computer Lab Safety Rules

* Know the location of the fire extinguisher and how to use them in case of an emergency.
* Report fires or accidents to your lecturer/laboratory technician immediately
* Report any broken plugs or exposed electrical wires to your lecturer/laboratory technician immediately.
* Avoid stepping on electrical wires or any other computer cables.
* Do not open the system unit casing or monitor casing particularly when the power is turned on.
* Do not touch, connect or disconnect any plug or cable without your lecturer/laboratory technician’s permission.
* Do not bring any food or drinks near the machine.