UNIVERSITY INSTITUTE OF ENGINEERING

&

TECHNOLOGY

MAHARISHI DAYANAND UNIVERSITY

PRACTICAL FILE

OBJECT ORIENTED PROGRAMMING

SUBMITTED TO:

DR. SUNITA DHINGRA

[ASST. PROFESSOR - CSE]

[UIET , MDU]

SUBMITTED BY: **SHUBHI**

CSE-1 , 23534

[UIET , MDU]

INDEX

PROGRAM-1

WRITE A PROGRAM THAT USES A CLASS WHERE THE MEMBER FUNCTIONS ARE DEFINED INSIDE A CLASS.

CODE:

main.cpp

```cpp
#include <iostream>
using namespace std;
class car
{
  private:
    int car_number;
    char car_model[10];
  public:
    void getdata()
     {
       cout<<"Enter car number: "; cin>>car_number;
       cout<<"\n Enter car model: "; cin>>car_model;
     }
    void showdata()
     {
       cout<<"Car number is "<<car_number;
       cout<<"\n Car model is "<<car_model;
     }
};
// main function starts
int main()
 {
    car c1;
    c1.getdata();
    c1.showdata();
    return 0;
 }
```

OUPTPUT:

```
Enter car number: 3546

 Enter car model: Martuti Suziki
Car number is 3546
 Car model is Martuti

...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-2

WRITE A PROGRAM THAT USES A CLASS WHERE THE MEMBER FUNCTIONS ARE DEFINED OUTSIDE A CLASS.

CODE:

```
/*Write a program that uses a class where the member functions are
defined outside a class.*/
#include <iostream>
using namespace std;
class product
{
    int number;
float price;

public:
    void getdata(int a, float b);
    void putdata();
};
void product ::getdata(int a, float b)
{
    number = a;
    price = b;
}
void product ::putdata()
{
    cout << "NUMBER : " << number << endl;
    cout << "PRICE :" << price << endl;
}
int main()
{
    product p;
    cout << "ENTER NUMBER:" << endl;
    int x;
    cin >> x;
    cout << "ENTER PRODUCT PRICE:" << endl;
    int y;
    cin >> y;
    p.getdata(x, y);
    p.putdata();
    return 0;
}
```

OUTPUT:

```
ENTER NUMBER:
26
ENTER PRODUCT PRICE:
789
NUMBER : 26
PRICE :789


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-3

WRITE A PROGRAM TO DEMONSTRATE THE USE OF STATIC DATA MEMBERS.

CODE:

```cpp
#include <iostream>
using namespace std;
class X
{
    int codeno;
    float price;
    static int count; //static data members
public:
    void getval(int i, float j)
    {
        codeno = i;
        price = j;
        ++count;
    }
    void display()
    {
        cout << "CODE NO:" << codeno << "\t";
        cout << "PRICE :" << price << endl;
    }
    static void dispcount()
    { //stactic member function
        cout << "COUNT= " << count << endl;
    }
};
int X ::count = 0;
int main()
{
    X ob1, ob2;
    ob1.getval(101, 89.99);
    ob2.getval(102, 99.99);
    X ::dispcount();         //static member function invoking
    X ob3;
    ob3.getval(103, 101.14);
    X ::dispcount();
    ob1.display();
    ob2.display();
    ob3.display();
    return 0;
}
```

OUTPUT:

```
COUNT= 2
COUNT= 3
CODE NO:101      PRICE :89.99
CODE NO:102      PRICE :99.99
CODE NO:103      PRICE :101.14


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-4

WRITE A PROGRAM TO DEMONSTRATE THE USE OF CONST DATA MEMBERS.

CODE:

main.cpp

```cpp
#include <iostream>
using namespace std;

class Number
{
private:
    const int x;

public:
    Number() : x(98) {} //const initialization
    void display()
    {
        cout << "x=" << x << endl;
    }
};
int main()
{
    Number NUM;
    NUM.display();

    return 0;
}
```

OUTPUT:

```
x=98


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-5

WRITE A PROGRAM TO DEMONSTRATE THE USE OF ZERO ARGUMENT AND PARAMETERIZED CONSTRUCTORS.

CODE:

main.cpp

```cpp
#include <iostream>
using namespace std;

class Test
{
    int a;
    char b;

public:
    Test()
    {
        cout << "This is default constructor with zero arguments" << endl;
    }
    Test(int i, char j)
    {
        a = i;
        b = j;
        cout << "Parameterized contstructor with two arguments: " << i << "\t" << j << endl;
    }
};
int main()
{
    Test tarray[6];
    Test newobj(56, 'n');
    return 0;
}
```

OUTPUT:

```
This is default constructor with zero arguments
This is default constructor with zero arguments
This is default constructor with zero arguments
This is default constructor with zero arguments
This is default constructor with zero arguments
This is default constructor with zero arguments
Parameterized contstructor with two arguments: 56       n


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-6

WRITE A PROGRAM TO DEMONSTRATE THE USE OF DYNAMIC CONSTRUCTOR.

```
#include <iostream>
#include <string.h>
using namespace std;
class String
{
    char *name;
    int size;

public:
    String()
    {
        size = 0;
        name = new char[size + 1];
    }
    String(char *s)
    {
        size = strlen(s);
        name = new char[size + 1];

        strcpy(name, s);
    }
    void display()
    {
        cout << name << endl;
    }
    void join(String &a, String &b);
};

void String ::join(String &a, String &b)
{
```

```
void String ::join(String &a, String &b)
{
    size = a.size + b.size;
    delete name;
    name = new char[size + 1];

    strcpy(name, a.name);
    strcat(name, b.name);
};
int main()
{
    char *first = "SHUBHI";
    String name1(first), name2("SHREYA"), name3("HARSH"), s1, s2;

    s1.join(name1, name2);
    s2.join(s1, name3);
    name1.display();
    name2.display();
    name3.display();
    s1.display();
    s2.display();


return 0;
}
```

OUTPUT:

```
SHUBHI
SHREYA
HARSH
SHUBHISHREYA
SHUBHISHREYAHARSH


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-7

WRITE A PROGRAM TO DEMONSTRATE THE OVERLOADING OF BINARY ARITHMETIC OPERATORS.

CODE:

```
#include <iostream>
using namespace std;
class binary
{
    float a1;
    float a2;
public:
    binary() {} //first constructor
    binary(float b1, float b2)
    {
        a1 = b1;
        a2 = b2;
    }
    binary operator-(binary);
    void display();
};
binary binary ::operator-(binary c)
{
    binary temp;
    temp.a1 = a1 - c.a1;
    temp.a2 = a2 - c.a2;
    return (temp);
}
void binary ::display()
{
    cout << a1 << " - j" << a2 << endl;
}
int main()
```

```
int main()
{
    binary c1, c2, c3;
    c1 = binary(7.5, 2.2);
    c2 = binary(9.6, 7.1);
    c3 = c1 - c2;

    cout << "c1 = ";
    c1.display();
    cout << "c2 = ";
    c2.display();
    cout << "c3 = ";
    c3.display();

    return 0;
}
```

OUTPUT:

```
c1 = 7.5 - j2.2
c2 = 9.6 - j7.1
c3 = -2.1 - j-4.9


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-8

WRITE A PROGRAM TO DEMONSTRATE THE MULTILEVEL INHERITANCE.

CODE:

```
#include <iostream>
#include <stdio.h>
using namespace std;
const int LEN = 25;
class PERSON
{
    char name[LEN];
    int age;

public:
    void readperson();
    void displayperson()
    {

        cout << "NAME: ";
        cout.write(name, LEN);
        cout << "\tAGE: " << age << endl;
    }
};
void PERSON ::readperson()
{
for (int i = 0; i < LEN; i++)
        name[i] = ' ';
    cout << "ENTER NAME OF THE PERSON: ";
    cin >> name;
    cout << "ENTER AGE : ";
    cin >> age;
}
//second class
class STUDENT : public PERSON
{
    int rollno;
    float average;
```

```
}
//second class
class STUDENT : public PERSON
{
    int rollno;
    float average;

public:
    void readstudent()
    {
        readperson();
        cout << "ENTER ROLL NO. : ";
        cin >> rollno;
        cout << "ENTER AVERAGE MARKS : ";
        cin >> average;
    }
    void disp_rollno()
    {
        cout << "ROLL NUMBER IS : " << rollno << endl;
    }
    float getaverage() { return average; }
};
class GRADSTUDENT : public STUDENT
{
    char subject[LEN];
    char working;
```

```
void GRADSTUDENT ::readit()
{
    readstudent();
    for (int i = 0; i < LEN; i++)
        subject[i] = ' ';
    cout << "ENTER MAIN SUBJECT: ";
    cin >> subject;
    cout << "WORKIN(Y/N): ? ";
    cin >> working;
}

int main()
{
    const int size = 5;
    GRADSTUDENT grad[size];
    int year, num_working = 0, non_working = 0, div1 = 0, total = 0;
    float topscore = 0, score, number, wperc, nwperc;
    cout << "ENTER YEAR : ";
    cin >> year;
    //loop for processing graduates info
    for (int i = 0; i < size; i++)
    {
        cout << "ENTER DETAILS FOR GRADUATE " << (i + 1) << endl;
        grad[i].readit();
        total++;
        //loop for counting working and non workings
        if ((grad[i].workstatus() == 'y') || (grad[i].workstatus() == 'Y'))
            num_working++;
        else
            non_working++;
        //determining top scorer
        score = grad[i].getaverage();
        if (score > topscore)
        {
            topscore = score;
```

```
            topscore = score;
            number = i;
        }
        //counting number of first divisioners
        if (score >= 60.0)
            div1++;
    }
    //Printing report
    int i = number;
    cout << endl
         << "\t\tREPORT OF THE YEAR " << year << endl;
cout << endl;
    cout << "WORKING GRADUATES : " << num_working << endl;
    cout << "NON WORKING GRADUATES: " << non_working << endl;
    cout << "\tDETAILS OF TOP SCORER" << endl;
    grad[i].displayperson();
    grad[i].displaysubject();
    nwperc = ((float)non_working / (float)total) * 100;
    wperc = ((float)div1 / (float)total) * 100;
    cout << "AVERAGE MARKS: " << grad[i].getaverage() << endl;
    cout << "\t " << nwperc << " % of the graduate this year are non_working and " << endl
         << wperc << " % are first divisioners" << endl;
    return 0;
}
```

OUTPUT:

```
ENTER YEAR : 2021
ENTER DETAILS FOR GRADUATE 1
ENTER NAME OF THE PERSON: Shubhi
ENTER AGE : 20
ENTER ROLL NO. : 23534
ENTER AVERAGE MARKS : 90
ENTER MAIN SUBJECT: Ma
char32_t                    keyword
WORKIN(Y/N): ? Y
ENTER DETAILS FOR GRADUATE 2
ENTER NAME OF THE PERSON: Harsh
ENTER AGE : 19
ENTER ROLL NO. : 23566
ENTER AVERAGE MARKS : 97
ENTER MAIN SUBJECT: Science
WORKIN(Y/N): ? Y
ENTER DETAILS FOR GRADUATE 3
ENTER NAME OF THE PERSON: Vaibhav
ENTER AGE : 23
ENTER ROLL NO. : 23587
ENTER AVERAGE MARKS : 96
ENTER MAIN SUBJECT: History
WORKIN(Y/N): ? Y

                REPORT OF THE YEAR 2021
```

```
ENTER DETAILS FOR GRA char32_t                    keyword
ENTER NAME OF THE PERSON: Vaibhav
ENTER AGE : 23
ENTER ROLL NO. : 23587
ENTER AVERAGE MARKS : 96
ENTER MAIN SUBJECT: History
WORKIN(Y/N): ? Y

                REPORT OF THE YEAR 2021

WORKING GRADUATES : 3
NON WORKING GRADUATES: 0
        DETAILS OF TOP SCORER
NAME: Harsh                     AGE: 19
SUBJECT : Science                 AVERAGE MARKS: 97
         0 % of the graduate this year are non_working and
100 % are first divisioners


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-9

WRITE A PROGRAM TO DEMONSTRATE THE MULTIPLE INHERITANCE.

CODE:

```
#include <iostream>
#include <stdio.h>
using namespace std;
class Base1
{
protected:
    int a;

public:
    Base1(int x)
    {
        a = x;
        cout << "CONSTRUCTING Base1" << endl;
    }
    ~Base1()
    {
        cout << "DESTRUCTING Base1" << endl;
    }
};
class Base2
{
protected:
    int b;
public:
    Base2(int y)
    {
        b = y;
        cout << "CONSTRUCTING Base2" << endl;
    }
    ~Base2()
    {
        cout << "DESTRUCTING Base2" << endl;
    }
};
class Derived : public Base2, public Base1
```

```
public:
    Derived(int i, int j, int k) : Base2(i), Base1(j)
    {
        c = k;
        cout << "CONSTRUCTING Derived" << endl;
    }
    ~Derived()
    {
        cout << "DESTRUCTING Derived" << endl;
    }
    void show()
    {
        cout << "1. " << a << endl
             << "2. " << b << endl
             << "3. " << c << endl;
    }
};
int main()
{
    Derived obj(14, 15, 16);
    obj.show();
    return 0;
}
```

OUTPUT:

```
CONSTRUCTING Base2
CONSTRUCTING Base1
CONSTRUCTING Derived
1. 15
2. 14
3. 16
DESTRUCTING Derived
DESTRUCTING Base1
DESTRUCTING Base2


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-10

WRITE A PROGRAM TO DEMONSTRATE THE VIRTUAL DERIVATION OF A CLASS.

CODE:

```
#include <iostream>
#include <stdlib.h>
using namespace std;
class Base
{
public:
    int a;
};
class B1 : virtual public Base
{
public:
    int b;
};
class B2 : virtual public Base
{
public:
    int c;
};
class B3 : public B1, public B2
{
public:
    int total;
};
int main()
{
    B3 obj;
    obj.a = 25;
    obj.b = 50;
    obj.c = 75;
    obj.total = obj.a + obj.b + obj.c;
    cout << obj.a << "\t\t" << obj.b << "\t\t" << obj.c << "\t\t" << obj.total << endl;
    return 0;
```

CODE:

```
input
25              50              75              150


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-11

WRITE A PROGRAM TO DEMONSTRATE THE USE OF FUNCTION TEMPLATE.

CODE:

```
#include <iostream>
using namespace std;
template <class T>
void bubbleSort(T a[], int n)
{
    for (int i = 0; i < n - 1; i++)
        for (int j = n - 1; i < j; j--)
            if (a[j] < a[j - 1])
                swap(a[j], a[j - 1]);
}
int main()
{
    int a[5] = {10, 50, 30, 40, 20};
    int n = sizeof(a) / sizeof(a[0]);
    bubbleSort<int>(a, n);

    cout << " Sorted array : ";
    for (int i = 0; i < n; i++)
        cout << a[i] << " ";
    cout << endl;

    return 0;
}
```

OUTPUT:

```
Sorted array : 10 20 30 40 50


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM-12

WRITE A PROGRAM TO DEMONSTRATE THE USE OF CLASS TEMPLATE

CODE:

```cpp
#include <iostream>
using namespace std;
template <class T>
class Calculator
{
private:
    T num1, num2;

public:
    Calculator(T n1, T n2)
    {
        num1 = n1;
        num2 = n2;
    }

    void displayResult()
    {
        cout << "Numbers are: " << num1 << " and " << num2 << "." << endl;
        cout << "Addition is: " << add() << endl;
        cout << "Subtraction is: " << subtract() << endl;
        cout << "Product is: " << multiply() << endl;
        cout << "Division is: " << divide() << endl;
    }

    T add() { return num1 + num2; }

    T subtract() { return num1 - num2; }

    T multiply() { return num1 * num2; }

    T divide() { return num1 / num2; }
};
```

```
    T add() { return num1 + num2; }

    T subtract() { return num1 - num2; }

    T multiply() { return num1 * num2; }

    T divide() { return num1 / num2; }
};

int main()
{
    Calculator<int> intCalc(2, 1);
    Calculator<float> floatCalc(2.4, 1.2);

    cout << "Int results:" << endl;
    intCalc.displayResult();

    cout << endl
         << "Float results:" << endl;
    floatCalc.displayResult();

    return 0;
}
```

OUTPUT:

```
Int results:
Numbers are: 2 and 1.
Addition is: 3
Subtraction is: 1
Product is: 2
Division is: 2

Float results:
Numbers are: 2.4 and 1.2.
Addition is: 3.6
Subtraction is: 1.2
Product is: 2.88
Division is: 2


...Program finished with exit code 0
```

THANK YOU!