OBJECT ORIENTED PROGRAMMING PRACTICAL

NAME – MRIDUL GUPTA

ENROLL NO. – 01916401518

SCHOOL – USICT

YEAR – 2ND YR.

B. TECH - IT

INDEX

BANK ACCOUNT

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

class Bank{
   char name_of_depositor[30];
   int Acc_no;
   char type_of_account[20];
   int Balance;

public:
   void Details(){
      cout<<"Enter your Name: ";
      cin>>name_of_depositor;
      cout<<"Account Number: ";
      cin>>Acc_no;
      cout<<"Type of Account: ";
      cin>>type_of_account;
      cout<<"Enter Balance: ";
      cin>>Balance;
   }

   void Deposit(){
      double deposit;
      cout<<"Enter the Amount to deposit: ";
      cin>>deposit;
      Balance += deposit;
   }

   void Withdraw(){
```

```
    double Amount;
    cout<<"Enter Amount to withdraw";
    cin>>Amount;
    if(Balance >= Amount)
        Balance -= Amount;
    else
        cout<<"Less Balance \n";
  }

  void Display(){
        cout<<"\t\t:Customer Details:\n\n";
        cout<<"Name: "<<name_of_depositor<<"\n";
        cout<<"Account No.: "<<Acc_no<<"\n";
        cout<<"Account Type: "<<type_of_account<<"\n";
        cout<<"Balance: "<<Balance<<"\n";
  }
};

int exit()
{
  return 0;
}

int main(){
  Bank b;
  int option;
  cout<<"YO YO Bank";
  cout<<"\n 1.Enter Details \n 2.Deposit \n 3.Withdraw \n 4.Display Details \n 5.Exit\n";
  do{
    cout<<"choose option: ";
```

```
        cin>>option;
        switch(option){
            case 1: b.Details();break;
            case 2: b.Deposit();break;
            case 3: b.Withdraw();break;
            case 4: b.Display();break;
            case 5: cout<<"Exitinig.....";break;
            default: cout<<"Enter valid option";
        }
    }while(option != 5);
    return 0;
}
```

OUTPUT:

```
YO YO Bank
 1.Enter Details
 2.Deposit
 3.Withdraw
 4.Display Details
 5.Exit
choose option: 1
Enter your Name: Mridul
Account Number: 125
Type of Account: Savings
Enter Balance: 500000
choose option: 4
                :Customer Details:

Name: Mridul
Account No.: 125
Account Type: Savings
Balance: 500000
choose option: 3
Enter Amount to withdraw500
choose option: 2
Enter the Amount to deposit: 10000
choose option: 4
                :Customer Details:

Name: Mridul
Account No.: 125
Account Type: Savings
Balance: 509500
choose option:
```

CALL BY VALUE

```
#include <iostream>
using namespace std;
void swap(float a,float b)
{
   float temp = a;
   a = b;
   b = temp;
   cout<<a<<" " <<b<<endl;
}

int main()
{
   float x, y;
   cout<<"Enter the values to be swapped"<<endl;
   cin>>x>>y;
   cout<<"Initial values at position 1 is "<<x<<" and at position 2 is "<<y<<endl;
   cout<<"After swapping the values are "<<endl;
   swap(x,y);
   return 0;
}
```

```
Enter the values to be swapped
5
3
Initial values at position 1 is 5 and at position 2 is 3
After swapping the values are
3 5
```

CALL BY REFERENCE

```
#include <iostream>

using namespace std;


void swap(float &a,float &b)

{

   float temp = a;

   a = b;

   b = temp;

}


int main()

{

   float x, y;

   cout << "Hello world!" << endl;

   cout<<"Enter the values to be swapped"<<endl;

   cin>>x>>y;

   cout<<"Initial values at position 1 is "<<x<<" and at position 2 is "<<y<<endl;

   cout<<"After swapping the values are "<<endl;

   swap(x,y);

   cout<<x<<" "<<y<<endl;

   return 0;

}
```

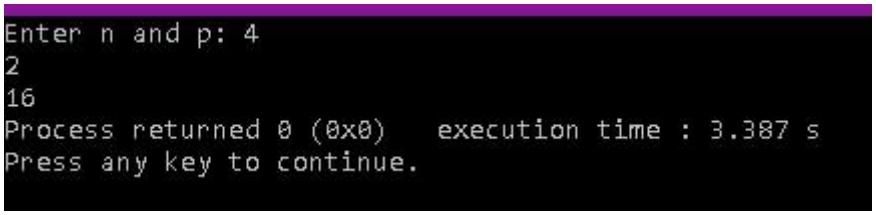
```
Enter the values to be swapped
5
7
Initial values at position 1 is 5 and at position 2 is 7
After swapping the values are
7 5

Process returned 0 (0x0)   execution time : 3.947 s
Press any key to continue.
```

POWER

```
#include<iostream>
using namespace std;
#include<math.h>


int power(int n,int p = 1){
   return pow(p,n);
}
int main(){
   int n,p;
   cout<<"Enter n and p: ";
   cin>>n>>p;
   cout<<power(n,p);
}
```



SORT AN ARRAY

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

void Sorting(int A[20], int _size)
{
   for(int i = 0; i<_size; i++)
   {
      for(int j = 0; j < _size - 1; j++)
      {
         if(A[i] < A[j])
         {
            int temp = A[i];
            A[i] = A[j];
            A[j] = temp;
         }
      }
   }
   for(int i = 0; i<_size; i++)
   {
      cout<<A[i]<<" ";
   }
}

int main()
{
   int A[20], _size;
   cout<<"Enter the Size of Array: ";
   cin>>_size;
   for(int i = 0; i < _size; i++)
```

```
    {
        cout<<(i+1)<<": ";
        cin>>A[i];
        cout<<"\n";
    }
    Sorting(A,_size);

    return 0;
}
```

```
Enter the Size of Array: 5
1: 64

2: 32

3: 1

4: 2

5: 33

1 2 32 33 64
Process returned 0 (0x0)   execution time : 10.497 s
Press any key to continue.
```

C ACCOUNT

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

class CAccount{
   int Acc_number;
   int Balance;
public:
   CAccount()
   {
      int AN, Bl;
      cout<<"Enter the Account No.: ";
      cin>>AN;
      cout<<"Enter the Balance: ";
      cin>>Bl;
      Acc_number = AN;
      Balance = Bl;
   }

   void InputTransaction(char Transaction_type, float Value)
   {
      if(Transaction_type == 'D')
      {
         Balance += Value;
      }
      else if(Transaction_type == 'W')
      {
         if(Value <= Balance)
            Balance -= Value;
         else
```

```
            cout<<"Sorry! Low Balance";
        }
    }

    void PrintBalance()
    {
        cout<<"Account Number: "<<Acc_number<<"\n";
        cout<<"Balance: "<<Balance;
    }
};

int main()
{
    int option, Value;
    char type;
    cout<<"\t\t.....CPP Bank.....\n\n";
    cout<<"1. Transaction \n2. Balance \n3. Exit \n\n";
    CAccount ca;
    do{
        cout<<"\nEnter a choice: ";
        cin>>option;
        switch(option)
        {
            case 1: cout<<"Enter the type of Transaction W or D: ";
                    cin>>type;
                    cout<<"\nEnter the Amount: ";
                    cin>>Value;
                    ca.InputTransaction(type, Value);
                    break;
            case 2: ca.PrintBalance();
```

```
            break;
        case 3: cout<<"Exiting";
            break;
        default: cout<<"Enter a valid Option";
      }
   }while(option != 3);
   return 0;
}
```

```
                .....CPP Bank.....

1. Transaction
2. Balance
3. Exit

Enter the Account No.: 1
Enter the Balance: 50000

Enter a choice: 2
Account Number: 1
Balance: 50000
Enter a choice: 1
Enter the type of Transaction W or D: W

Enter the Amount: 500

Enter a choice: 2
Account Number: 1
Balance: 49500
Enter a choice: 1
Enter the type of Transaction W or D: D

Enter the Amount: 3000

Enter a choice: 2
Account Number: 1
Balance: 52500
Enter a choice: _
```

DISTANCE

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

class Distance
{
   float feet;
   float inches;
public:
   Distance() //Blank Constructor
   {
      feet = 0.0;
      inches = 0.0;
   }

   Distance(float f, float i) //Parameterized Constructor
   {
      feet = f;
      inches = i;
   }

   void Input_details() //Input from the User
   {
      cout<<"Enter the feet: ";
      cin>>feet;
      cout<<"Enter the inches: ";
      cin>>inches;
   }

   void Adjust() //Adjust The Inches
```

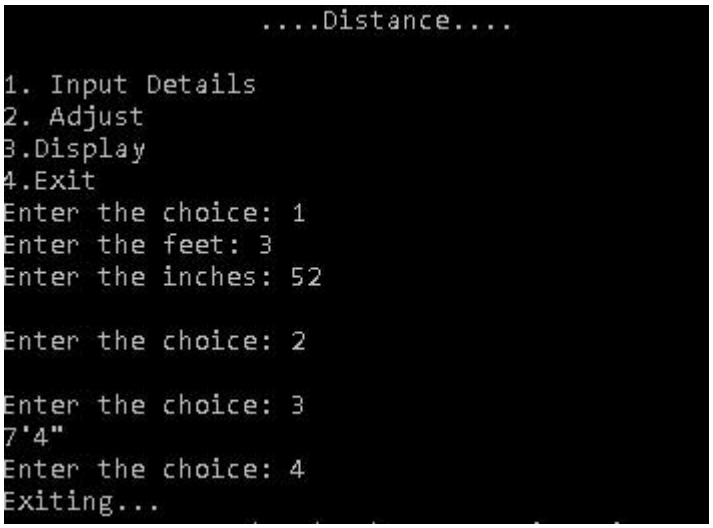
```
    {
        if(inches >= 12)
        {
            feet += (int)inches/12;
            inches = (int)inches%12;
        }
    }

    void Display()
    {
        cout<<feet<<"'"<<inches<<"\"";
    }
};

int main()
{
   Distance d;
   int option;
   cout<<"\t\t....Distance....\n\n";
   cout<<"1. Input Details \n2. Adjust \n3.Display \n4.Exit";
   do{
        cout<<"\nEnter the choice: ";
        cin>>option;

        switch(option)
        {
            case 1: d.Input_details();
                    break;
            case 2: d.Adjust();
                    break;
```

```
        case 3: d.Display();
                break;
        case 4: cout<<"Exiting...";
                break;
        default: cout<<"Enter a valid option";
      }
  }while(option != 4);
  return 0;
}
```



SEQUENCE OF NAMES

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

void sort_names(char A[20][20], int _size)
{
   char temp[20];
   for(int i = 0; i<_size; i++)
   {
      for(int j = 0; j<_size - 1; j++)
      {
         if(strlen(A[i]) < strlen(A[j]))
         {
            strcpy(temp,A[i]);
            strcpy(A[i],A[j]);
            strcpy(A[j],temp);
         }
      }
   }

   for(int i = 0; i < _size; i++)
   {
      cout<<A[i]<<" ";
   }
}


int main()
{
```

```
    int _size;
    char A[20][20];
    cout<<"Enter the size: ";
    cin>>_size;
    for(int i = 0; i < _size; i++)
    {
       cin>>A[i];
    }
    sort_names(A, _size);
    return 0;
}
```

```
Enter the size: 5
abcdef
abc
a
ab
abcd
a ab abc abcd abcdef
```

STRING

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

class String{
      char *p = new char;

   public:
      String(){
         p = "Hi";
      }

      String(const String &object){
         p = object.p;
      }

      ~String(){
         delete p;
      }

      void input_details(char val){
         *p = val;
      }

      void display(){
         cout<<"String Entered: "<<*p;
      }

      char rev(){
```

```
        char r[20];
        int j = strlen(p)-1;
        for(int i = 0; p[i]; i++){
           r[j--] = p[i];
        }
        return *r;
      }
};

int main(){
   String s;
   char a[] = "Hello";
   s.input_details(*a);
   s.display();
   char r = s.rev();
   cout<<"Reverse String: "<<r;
   return 0;
}
```

```
String Entered: Hello
Reverse String: olleH
Process returned 0 (0x0)   execution time : 0.190 s
Press any key to continue.
```

DISTANCEFI AND DISTANCEMC

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

class DistanceMC;

class DistanceFI
{
private:
   float feet = 0;
   float inches = 0;
public:
   void input_details(float f, float I)
   {
      feet = f;
      inches = I;
   }

   void Display()
   {
      cout<<feet<<"' "<<inches<<'"'<<"\n";
   }

   DistanceFI operator +(DistanceFI &obj)
   {
      DistanceFI d;
      d.feet = this->feet + obj.feet;
      d.inches = this->inches + obj.inches;
      return d;
   }
```

```
    friend const DistanceFI operator+( const DistanceFI& m, const DistanceMC& n);
    friend ostream& operator<<(ostream& os, const DistanceFI& c);
};

ostream& operator<<(ostream &os, const DistanceFI& c)
 {
    os << c.feet << c.inches<<"\n";
    return os;
}

class DistanceMC
{
private:
    float metres = 0;
    float centimetres = 0;
public:
    void input_details(float m, float c)
    {
        metres = m;
        centimetres = c;
    }

    void Display()
    {
        cout<<metres<<"m "<<centimetres<<"cm"<<"\n";
    }

    DistanceMC operator +(DistanceMC &obj)
    {
        DistanceMC d;
```

```
        d.metres = this->metres + obj.metres;
        d.centimetres = this->centimetres + obj.centimetres;
        return d;
    }
    friend const DistanceFI operator+( const DistanceFI& m, const DistanceMC& n);
};

const DistanceFI operator+(const DistanceFI& c1, const DistanceMC& c2)
{
     DistanceFI temp;
     temp.feet = c1.feet + 3.28084*c2.metres;
     temp.inches = c1.inches + 0.393701*c2.centimetres;
     return temp;
 }

int main()
{
    DistanceFI fi;
    DistanceMC mc;
    fi.input_details(10,5);
    fi.Display();
    mc.input_details(20,4);
    mc.Display();
    DistanceFI f = fi + mc;
    f.Display();
    return 0;
}
```

```
10' 5"
20m 4cm
75.6168' 6.5748"

Process returned 0 (0x0)   execution time : 0.289 s
Press any key to continue.
```

COMPLEX

```
#include<iostream>
#include<math.h>
using namespace std;

class Complex_Number
{
private:
   float real;
   float img;
public:

   Complex_Number()
   {
      real = 0;
      img = 0;
   }

   void Input(float r, float i)
   {
      real = r;
      img = i;
   }

   void Display()
   {
      cout<<real<<" + i"<<img;
   }

   Complex_Number operator +(Complex_Number &obj)
```

```
{
    Complex_Number c;
    c.real = this->real + obj.real;
    c.img = this->img + obj.img;
    return c;
}

Complex_Number operator -(Complex_Number &obj)
{
    Complex_Number c;
    c.real = this->real - obj.real;
    c.img = this->img - obj.img;
    return c;
}

// (a + ib)*(c + id) = (ac - bd) + i(bc + ad)
Complex_Number operator *(Complex_Number &obj)
{
    Complex_Number c;
    c.real = this->real * obj.real - this->img * obj.img;
    c.img = this->img * obj.real + this->real * obj.img;
    return c;
}

// (a + ib)/(c + id) = (ac + bd)/(c^2 + d^2) + (bc - ad)/(c^2 + d^2)
Complex_Number operator /(Complex_Number &obj)
{
    Complex_Number c;
    c.real = (this->real * obj.real + this->img * obj.img)/(pow(obj.real, 2) + pow(obj.img, 2));
    c.img = (this->img * obj.real - this->real * obj.img)/(pow(obj.real, 2) + pow(obj.img, 2));
```

```
        return c;
    }
};

int main()
{
    Complex_Number c1, c2;
    c1.Input(5,4);
    c2.Input(6,1);

    Complex_Number c3 = c1+c2;
    c1.Display();
    cout<<" + ";
    c2.Display();
    cout<<" = ";
    c3.Display();
    cout<<"\n";

    c3 = c1-c2;
    c1.Display();
    cout<<" - ";
    c2.Display();
    cout<<" = ";
    c3.Display();
    cout<<"\n";

    c3 = c1*c2;
    c1.Display();
    cout<<" * ";
    c2.Display();
```

```
    cout<<" = ";
    c3.Display();
    cout<<"\n";

    c3 = c1/c2;
    c1.Display();
    cout<<" / ";
    c2.Display();
    cout<<" = ";
    c3.Display();
}
```

```
5 + i4 + 6 + i1 = 11 + i5
5 + i4 - 6 + i1 = -1 + i3
5 + i4 * 6 + i1 = 26 + i29
5 + i4 / 6 + i1 = 0.918919 + i0.513514
Process returned 0 (0x0)   execution time : 0.216 s
Press any key to continue.
```

VECTOR

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

class Vector_
{
private:
   float arr[10];
   int index = 0;
public:
   void modify(int n, int value)
   {
      arr[n-1] = value;
   }

   Vector_ operator *(int scalar)
   {
      Vector_ v;
      for(int i = 0; i < index; i++)
      {
         v.arr[v.index++] = scalar*arr[i];
      }
      return v;
   }

   void operator >>(float a)
   {
      arr[index++] = a;
   }
```

```
    friend ostream& operator<<(ostream& os, const Vector_& v);
};

ostream& operator<<(ostream& os, const Vector_& v)
{
    cout<<"{ ";
    for(int i = 0; i <v.index; i++)
    {
        cout<<v.arr[i]<<", ";
    }
    cout<<"}";
    return os;
}

int main()
{
    Vector_ v;
    v>>3;
    v>>4;
    v>>5;
    v>>6;
    v>>7;
    cout<<v;
    v.modify(2,9);
    cout<<"\nModified\n";
    cout<<v;
    cout<<"\nMultiplied with 2\n";
    Vector_ v2 = v*2;
    cout<<v2;
    return 0;
```

}

```
{ 3, 4, 5, 6, 7, }
Modified
{ 3, 9, 5, 6, 7, }
Multiplied with 2
{ 6, 18, 10, 12, 14, }
Process returned 0 (0x0)   execution time : 0.200 s
Press any key to continue.
```

HIERARCHY

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

class Person
{
   string Name;
   int Phone_num;
   int std_id;
   int Emp_id;
public:
   Person()
   {
      Phone_num = 0;
      std_id = 0;
      Emp_id = 0;
      Name = '__';
   }

   void Input_details(string name, int ph_num, int s_id = 0, int e_id = 0)
   {
      Name = name;
      Phone_num = ph_num;
      std_id = s_id;
      Emp_id = e_id;
   }

   void Display()
   {
```

```
        cout<<"Name: "<<Name<<"\n";
        cout<<"Phone Number: "<<Phone_num<<"\n";
        cout<<"Employee Id: "<<Emp_id<<"\n";
        cout<<"Student Id: "<<std_id<<"\n";
    }
};

class Employee:public Person
{
    Person p;
public:
    void Input_details(string name, int ph_num, int e_id)
    {
        p.Input_details(name,ph_num, 0,e_id);
    }

    void Display()
    {
        p.Display();
    }
};

class Student:public Person
{
    Person p;
public:

    void Input_details(string name, int ph_num, int s_id)
    {
        p.Input_details(name,ph_num,s_id,0);
```

```
    }

    void Display()
    {
        p.Display();
    }
};

int main()
{
    Employee e;
    e.Input_details("Mridul Gupta",93543854,1);
    e.Display();
    cout<<"\n";

    Student s;
    s.Input_details("Mridul Gupta",9354385487,1);
    s.Display();
    return 0;
}
```

```
Name: Mridul Gupta
Phone Number: 93543854
Employee Id: 1
Student Id: 0

Name: Mridul Gupta
Phone Number: 764450895
Employee Id: 0
Student Id: 1

Process returned 0 (0x0)   execution time : 0.276 s
Press any key to continue.
```

SHAPE

```
#include<iostream>
#include<stdlib.h>
using namespace std;

class Shape
{
public:
   double length;
   double height;
};

class Triangle:public Shape
{
   Shape triangle;

public:
   void Input_details(double l, double h)
   {
      triangle.length = l;
      triangle.height = h;
   }

   void Area()
   {
      double area;
      area = 0.5 * triangle.length * triangle.height;
      cout<<"\nArea of Triangle: "<<area<<"\n";
   }
};
```

```
class Rectangle:public Shape
{
   Shape rectangle;

public:
   void Input_details(double l, double h)
   {
      rectangle.length = l;
      rectangle.height = h;
   }

   void Area()
   {
      double area;
      area = rectangle.length * rectangle.height;
      cout<<"\nArea of Rectangle: "<<area<<"\n";
   }
};

int main()
{
   int ch;
   cout<<"\t\tShape\n\n";
   cout<<"1. Triangle\n2. Rectangle\n3. Exit";
   do
   {
      cout<<"\nChoose an option: ";
   cin>>ch;
   switch(ch)
```

```
    {
        case 1: double b,h;
                system("CLS");
                cout<<"Enter the dimension base - height: \n";
                cin>>b>>h;
                Triangle t;
                t.Input_details(b,h);
                t.Area();
                break;
        case 2: double l,w;
                system("CLS");
                cout<<"Enter the dimension length - breadth: \n";
                cin>>l>>w;
                Rectangle r;
                r.Input_details(l,w);
                r.Area();
                break;
        case 3: cout<<"Exiting....";break;
        default: cout<<"Enter a Valid Option";
    }
    }while(ch != 3);
    return 0;
}
```

```
Enter the dimension length - breadth:
6
8

Area of Rectangle: 48

Choose an option:
```

```
                Shape

1. Triangle
2. Rectangle
3. Exit
Choose an option:
```