Rollwala Computer Center

OBJECT ORIENTED CONCEPTS AND PROGRAMMING

Assignment-1

Feb 02, 2022

Name: **Akshit Trivedi**

Roll No: **40**

Course: **Master of Computer Application**

Sem: **1**

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

1. **Write a program that reads a character and prints out whether it is a vowel or a consonant or a number or any other character.**

INPUT:

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

int main()
{
    char c;
    cout<<"Enter any  Character: ";
    cin>>c;
    c=(char)tolower(c);
    cout<<"Character Entered by you is: "<<c<<endl;
    if(isdigit(c))
    {
        cout<<"\n"<<c<<" is a Digit.";
    }
    else if(isalpha(c))
    {
        if(c=='a' || c=='e' || c=='i' || c=='o' || c=='u')
        {
            cout<<"\n"<<c<<" is a Vowel.";
        }
        else
        {
            cout<<"\n"<<c<<" is a Constant.";
        }
    }
    else
    {
        cout<<"\n"<<c<<" is a Special Character.";
    }
    return 0;
}
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**
Class: MCA-1 **Year: 2021-22**

OUTPUT:

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\1_Constant_Vowel.exe
Enter any Character: a
Character Entered by you is: a

a is a Vowel.
```

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\1_Constant_Vowel.exe
Enter any Character: *
Character Entered by you is: *

* is a Special Character.
```

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\1_Constant_Vowel.exe
Enter any Character: 8
Character Entered by you is: 8

8 is a Digit.
```

2. WAP to add an 8% tax to a given amount and round the net amount to its positive nearest amount.

INPUT:

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


int main()
{
    float amt, tax;
    int tot_amt;
    cout<<"Enter the Amount: ";
    cin>>amt;
```

```
    tax = amt * 0.08;
    tot_amt = int(amt + tax);
    cout<<"The Tax on amount: "<< amt << " is " << tax << endl;
    cout<<"The Total amount is: "<< tot_amt;
    return 0;
}
```

OUTPUT:

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\2_Tax.exe
Enter the Amount: 25000
The Tax on amount: 25000 is 2000
The Total amount is: 27000
```

4. WAP that takes a series of numbers and counts the frequency of positive values and negative values.

INPUT:

```
#include<iostream>
using namespace std;
int main()
{
    int pos=0,neg=0,zero=0,arr[100],i,arr_size;
    cout<<"Enter the How Many Elements you want to Enter (Max 100): ";
    cin>>arr_size;
    cout<<"\nEnter "<<arr_size<<" Elements: ";
    for(i=0;i<arr_size;i++)
    {
      cout<<"\nEnter Element "<<i+1<<" : ";
        cin>>arr[i];
```

```
        if(arr[i]<0)
        {
            neg++;
        }
        else if(arr[i]==0)
        {
            zero++;
        }
        else
        {
            pos++;
        }
    }
    cout<<"\nFrequency of Positive Numbers: "<<pos;
    cout<<"\nFrequency of Negative Numbers: "<<neg;
    cout<<"\nFrequency of Zero: "<<zero;
    return 0;
}
```

OUTPUT:

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\4_Count_Positive_Negative.exe
Enter the How Many Elements you want to Enter (Max 100): 5

Enter 5 Elements:
Enter Element 1 : -5

Enter Element 2 : 40

Enter Element 3 : -16

Enter Element 4 : 0

Enter Element 5 : 21

Frequency of Positive Numbers: 2
Frequency of Negative Numbers: 2
Frequency of Zero: 1
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

5. WAP to convert the distance in meter to centimeter and feet to inches and vice versa using class DISTANCE. (1 meter = 100 centimeter and 1 feet = 12 inches).

INPUT:

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

class Distance
{
    public:
    float met_to_cen(float meter)
    {
        return  meter * 100;
    }

    float cen_to_met(float centi)
    {
        return centi / 100;
    }

    float feet_to_inch(float feet)
    {
        return feet * 12;
    }

    float inch_to_feet(float inch)
    {
```

```
        return inch / 12;
    }
};

int main()
{
    Distance d1;
    int op = 0;
    float item;
    while(op != 5)
    {
        cout<<"\n1: Convert from METRES To CENTIMETRES"<<endl;
        cout<<"2: Convert from CENTIMETRES To METRES"<<endl;
        cout<<"3: Convert from FEET To INCHES"<<endl;
        cout<<"4: Convert from INCHES To FEET"<<endl;
        cout<<"5: Exit"<<endl;
        cout<<"Enter your choice: ";
        cin>>op;

            cout<<"\n";
        switch(op)
        {
            case 1: cout<<"Enter the input in meter: ";
                    cin>>item;
                    cout<<item <<" Metres = " << d1.met_to_cen(item) << "
Centimetres"<< endl;
                    break;
            case 2: cout<<"Enter the input in centimeter: ";
```

```
                    cin>>item;
                    cout<< item <<" Centimetres = " << d1.cen_to_met(item) <<
" Metres"<< endl;
                    break;
            case 3: cout<<"Enter the input in feet: ";
                    cin>>item;
                    cout<< item <<" Feet = " << d1.feet_to_inch(item) << "
Inches"<< endl;
                    break;
            case 4: cout<<"Enter the input in inches: ";
                    cin>>item;
                    cout<< item <<" Inches = " << d1.inch_to_feet(item) << "
Feet"<< endl;
                    break;
            case 5: cout<<"Exiting the program" << endl;
                    break;
            default : cout<<"Enter a valid choice" << endl;
                    break;
        }
    }
}
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

OUTPUT:

```
Select C:\MCA\MCA-1\OOCP\Practical Assignment 1\5_Distance_Convertor.exe

1: Convert from METRES To CENTIMETRES
2: Convert from CENTIMETRES To METRES
3: Convert from FEET To INCHES
4: Convert from INCHES To FEET
5: Exit
Enter your choice: 1

Enter the input in meter: 153
153 Metres = 15300 Centimetres

1: Convert from METRES To CENTIMETRES
2: Convert from CENTIMETRES To METRES
3: Convert from FEET To INCHES
4: Convert from INCHES To FEET
5: Exit
Enter your choice: 2

Enter the input in centimeter: 15300
15300 Centimetres = 153 Metres

1: Convert from METRES To CENTIMETRES
2: Convert from CENTIMETRES To METRES
3: Convert from FEET To INCHES
4: Convert from INCHES To FEET
5: Exit
Enter your choice: 3

Enter the input in feet: 123
123 Feet = 1476 Inches

1: Convert from METRES To CENTIMETRES
2: Convert from CENTIMETRES To METRES
3: Convert from FEET To INCHES
4: Convert from INCHES To FEET
5: Exit
Enter your choice: 4

Enter the input in inches: 1476
1476 Inches = 123 Feet

1: Convert from METRES To CENTIMETRES
2: Convert from CENTIMETRES To METRES
3: Convert from FEET To INCHES
4: Convert from INCHES To FEET
5: Exit
Enter your choice: 5

Exiting the program
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

6. Create a class for Bank account with the following data members.

(1) Name of depositor

(2) Account number

(3) Type of account

(4) Balance member functions

a. To assign initial values

b. To deposit an amount in a particular account

c. To withdraw an amount after checking the balance

d. To display name and balance

WAP to manage at least 10 customers who can deal with deposit and withdraw amount and calculate the current balance.

INPUT:

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

class Account
{
    //data members
    static int totAccount;
    char name[20];
    int account_no;
    char account_type[20];
    float bal=0;

    public:
    Account()
    {
```

```
        account_no = 1000 + totAccount;
        totAccount++;
        bal=0;
    }

    //member function
    void deposit();
    void withdraw();
    void show_bal();
    void addAccount();

};

int Account::totAccount=0;

void Account::addAccount()
{
        int choice=0;
        cout<<"Enter your name: ";
        cin>>name;

        while(choice != 1 && choice != 2)
        {
            cout<<"Please choose your Account Types"<<endl;
            cout<<"1: Saving Account"<<endl;
            cout<<"2: Current Account"<<endl;
            cout<<"Enter your choice: ";
            cin>>choice;
```

```
            switch(choice)
            {
                case 1: strcpy(account_type, "Saving Account");
                    break;
                case 2: strcpy(account_type, "Current Account");
                    break;
                default: cout<<"Please enter a valid choice"<<endl;
                    break;
            }

        }
            cout<<"\n\nFinal details:"<<endl;
            cout<<"Account Number: "<<account_no<<endl;
            cout<<"Name: "<<name<<endl;
            cout<<"Account Type: "<<account_type<<endl;
            cout<<"Your initial balance is 0.Please deposit into your
account\n\n";

}

void Account::deposit()
{
    int cash=0;
    cout<<"Enter the amount: ";
    cin>>cash;
    bal = bal + cash;
    cout<<"The balance has been updated"<<endl;
```

```
}


void Account::withdraw()
{
    int cash=0;
    cout<<"Enter the amount: ";
    cin>>cash;
    if(bal==0 || cash>bal)
    {
        cout<<"You don't have sufficient balance to withdraw the given
amount"<<endl;
        return;
    }
    bal = bal - cash;
    cout<<"Your current balance is: "<<bal<<endl;
}


void Account::show_bal()
{
    cout<<"Account_no: "<< account_no << endl;
    cout<<"Name: "<< name << endl;
    cout<<"Total Balance: " << bal<<endl;
}


int check_account_no()
{
    int account_no;
    cout<<"Enter your account number: ";
```

```
    cin>>account_no;
    int j = account_no % 1000;
    if(j>10 || j<0 || account_no > 1011 || account_no < 1000 )
    {
        cout<<"Invalid Account Number"<<endl;
        return -1;
    }
    return j;
}

int main()
{
    int i=0;
    Account account[5];
    int accno, j;
    int op;
    while(op!=5)
    {
        cout<<"\n1: Add a new Account"<<endl;
        cout<<"2: Deposit"<<endl;
        cout<<"3: Withdraw"<<endl;
        cout<<"4: Show details"<<endl;
        cout<<"5: Exit"<<endl;
        cout<<"Enter your choice: ";
        cin>>op;
        switch(op)
        {
            case 1:
```

```
                if(i==5)
                {
                    cout<<"Users are exceeded"<<endl;
                    break;
                }
                account[i++].addAccount();
                break;

        case 2:
                j = check_account_no();
                if(j==-1)
                {
                    break;
                }
                account[j].deposit();
                break;

        case 3:
                j = check_account_no();
                if(j==-1)
                {
                    break;
                }
                account[j].withdraw();
                break;

        case 4:
                j = check_account_no();
```

```
                    if(j==-1)
                    {
                        break;
                    }
                    account[j].show_bal();
                    break;

            case 5: cout<<"Exiting!!!"<<endl;
                    break;

            default: cout<<"Please enter a valid choice";
                    break;
        }
    }
    return 0;
}
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

OUTPUT:

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

C:\MCA\MCA-1\OOCP\Practical Assignment 1\6_Bank_Account.exe

```
1: Add a new Account
2: Deposit
3: Withdraw
4: Show details
5: Exit
Enter your choice: 1
Enter your name: Akshit
Please choose your Account Types
1: Saving Account
2: Current Account
Enter your choice: 1


Final details:
Account Number: 1000
Name: Akshit
Account Type: Saving Account
Your initial balance is 0.Please deposit into your account


1: Add a new Account
2: Deposit
3: Withdraw
4: Show details
5: Exit
Enter your choice: 2
Enter your account number: 1000
Enter the amount: 5000
The balance has been updated

1: Add a new Account
2: Deposit
3: Withdraw
4: Show details
5: Exit
Enter your choice: 4
Enter your account number: 1000
Account_no: 1000
Name: Akshit
Total Balance: 5000

1: Add a new Account
2: Deposit
3: Withdraw
4: Show details
5: Exit
Enter your choice: 3
Enter your account number: 1000
Enter the amount: 1000
Your current balance is: 4000
```

```
1: Add a new Account
2: Deposit
3: Withdraw
4: Show details
5: Exit
Enter your choice: 4
Enter your account number: 1000
Account_no: 1000
Name: Akshit
Total Balance: 4000

1: Add a new Account
2: Deposit
3: Withdraw
4: Show details
5: Exit
Enter your choice: 5
Exiting!!!
```

7. WAP to display taxi details, Customer name and total fare that must be calculated as:
For first 5 km, fare is 50 rs., for next 10 kms, fare is 12 rs./km, for next 15 kms fare is 8 rs./km and 5 rs./km for more than 25 kms.

INPUT:

```
#include<iostream>

using namespace std;

int main()

{

    int fare=0,km,taxi_no;

    string cust_name;

    cout<<"Enter Taxi No: ";

    cin>>taxi_no;

    cout<<"Enter Customer Name: ";

    cin>>cust_name;

    cout<<"Enter how many KM: ";

    cin>>km;

    if(km<=5)
```

```
    {
        fare=50;
    }
    else if(km>5 && km<=15)
    {
        fare=50+(km-5)*12;
    }
    else if(km>15 && km<=30)
    {
        fare=50+120+(km-15)*8;
    }
    else if(km>30)
    {
        fare=50+120+120+(km-30)*5;
    }
    cout<<"\n---BILL---\n";
    cout<<"\nTaxi No is: "<<taxi_no;
    cout<<"\nCustomer Name is: "<<cust_name;
    cout<<"\nYour Fare is: "<<fare;
    return 0;
}
```

OUTPUT:

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\7_Taxi_Fair.exe
Enter Taxi No: 40
Enter Customer Name: Akshit
Enter how many KM: 500

---BILL---

Taxi No is: 40
Customer Name is: Akshit
Your Fare is: 2640
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

8. Write a program to display time in hours, minutes and seconds after adding integer value to the time type object, show the use of constructors.

INPUT:

```
#include<iostream>
#include<conio.h>

using namespace std;

class Time{
    int hours, minutes, seconds;
    public:
    Time()
    {

    }
    Time(int hrs, int min, int sec)
    {
        hours = hrs;
        minutes = min;
        seconds = sec;
    }
    void show()
    {
        cout<<this->hours<<":"<<this->minutes<<":"<<this->seconds<<endl;
    }

};
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

```
int main()
{
    int h, m, s;
    cout<<"\n Enter Hours input: ";
    cin>>h;
    cout<<"\n Enter Minites input: ";
    cin>>m;
    cout<<"\n Enter Seconds input: ";
    cin>>s;

    Time time(h, m, s);
    cout<<"\n Output is : ";

    time.show();
    return 0;
}
```

OUTPUT:

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\8_Time_Const.exe

Enter Hours input: 7

Enter Minites input: 15

Enter Seconds input: 30

Output is : 7:15:30
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

9. Construct the class named person (Data member: age, name), Write member functions: detail(), display(). Write a program that will accept the detail of four persons and create function such that it will find the max age from the supplied detail.

INPUT:

```
#include<iostream>
using namespace std;
class Person{
public:
	string name;
	float age;
public:
	Person(){


	}
	Person(string n, float g){
		name = n;
		age = g;
	}
	void detail(){
		cout<<"Enter the person name: ";
		cin>>name;
		cout<<"Enter the person age: ";
		cin>>age;
	}
	void display(){
		cout<<"\n\nName is: "<<name;
		cout<<"\nAge is: "<<age;
```

```
	}
	friend void max_age(Person p1, Person p2, Person p3, Person p4);
};

void max_age(Person p1, Person p2, Person p3, Person p4)
{
	if(p1.age > p2.age && p1.age > p3.age && p1.age > p4.age)
	{
		cout<<"\n\nMaximum age is: "<<p1.age;
		cout<<"\nThe name of the person: "<<p1.name;
	}
	else if(p2.age > p1.age && p2.age > p3.age && p2.age > p4.age)
	{
		cout<<"\n\nMaximum age is: "<<p2.age;
		cout<<"\nThe name of the person: "<<p2.name;
	}
	else if(p3.age > p1.age && p3.age > p2.age && p3.age > p4.age)
	{
		cout<<"\n\nMaximum age is: "<<p3.age;
		cout<<"\nThe name of the person: "<<p3.name;
	}
	else
	{
		cout<<"\n\nMaximum age is: "<<p4.age;
		cout<<"\nThe name of the person: "<<p4.name;
	}
}
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

```
int main(){

	Person t1,t2,t3,t4;


	t1.detail();
	t2.detail();
	t3.detail();
	t4.detail();
	cout<<"\nDisplaying Data:-";
	t1.display();
	t2.display();
	t3.display();
	t4.display();
	max_age(t1, t2, t3, t4);
}
```

OUTPUT:

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\9_Class_Person_Display_Detail.exe
Enter the person name: Akshit
Enter the person age: 21
Enter the person name: Sagar
Enter the person age: 22
Enter the person name: Sijo
Enter the person age: 23
Enter the person name: Yash
Enter the person age: 25

Displaying Data:-

Name is: Akshit
Age is: 21

Name is: Sagar
Age is: 22

Name is: Sijo
Age is: 23

Name is: Yash
Age is: 25

Maximum age is: 25
The name of the person: Yash
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

11. Create class DATE having date, month and year as data members. Update the date with user given days using friend function, and display new date.

INPUT:

```
#include <iostream>
using namespace std;
class Date
{
public:
    int date;
    int month;
    int year;
    friend void set(int, Date &);
    void get(Date date)
    {
        cout << "Date:" << date.date << endl;
        cout << "Month:" << date.month << endl;
        cout << "Year:" << date.year;
    }
};
void set(int days, Date &date)
{
    date.year = days / 365;
    date.month = (days % 365) / 30;
    date.date = (days % 365) % 30;
}
int main()
{
    Date date;
    int days;
    cout << "Enter days:";
    cin >> days;
    set(days, date);
    date.get(date);
    return 0;
}  }
    return 0;
}
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

OUTPUT:

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\11_Date.exe
Enter days: 35
Date: 5
Month: 1
Year: 0
```

12. Create class VECTOR (int x, int y, int z). Using parameterized constructor with default arguments, initialize the data members. Also using member function add two objects of this class and display resultant value using member functions.

INPUT:

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

class VECTOR
{
    int x;
    int y;
    int z;

    public:
    VECTOR(int x=0, int y=0, int z=0)
    {
        this->x = x;
        this->y = y;
        this->z = z;
    }

    void addingvectors(VECTOR other)
    {
        cout<<"X: "<<x + other.x<<endl;
        cout<<"Y: "<<y + other.y<<endl;
        cout<<"Z: "<<z + other.z<<endl;
    }
```

```
};

int main()
{
    VECTOR v1(25);
    VECTOR v2(25,75,100);
    v1.addingvectors(v2);
    return 0;
}
```

OUTPUT:

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\12_Vector.exe
X: 50
Y: 75
Z: 100
```

13. Design an airline reservation data structure that contains the following data:

Flight number
Originating airport code (3 characters)
Destination airport code (3 characters)
Departure time
Arrival time

Write a program that lists all the planes that leave from two airports specified by the user.

INPUT:

```
#include <iostream>

#include <string>

using namespace std;

class Flight

{

private:

    string flightNo;
```

```
    string originated;
    string destination;
    string departure;
    string arrival;


public:
    void set_details(string flightNo, string originated, string destination,
string departure, string arrival)
    {
        this->flightNo = flightNo;
        this->originated = originated;
        this->destination = destination;
        this->departure = departure;
        this->arrival = arrival;
    }
    void get_details()
    {
        cout << "Flight no: " << this->flightNo << endl;
        cout << "Originating Airport Code: " << this->originated << endl;
        cout << "Destination Airport Code: " << this->destination << endl;
        cout << "Departure time: " << this->departure << endl;
        cout << "Arrival time: " << this->arrival << endl;
    }
};


int main()
{
    int n;
```

```
    cout << "Enter number of flights: ";
    cin >> n;
    Flight flights[n];
    for (int i = 0; i < n; i++)
    {
        string flightNo, originated, destination, departure, arrival;
        cout << "\nFlight Number: " << i + 1 << " \n";
        cout << "Enter Flight no: ";
        cin >> flightNo;
        cout << "Enter Originating Airport code: ";
        cin >> originated;
        cout << "Enter Destination Airport code: ";
        cin >> destination;
        cout << "Enter departure time: ";
        cin >> departure;
        cout << "Enter Arrival time: ";
        cin >> arrival;
        flights[i].set_details(flightNo, originated, destination, departure,
arrival);
    }
    for (int i = 0; i < n; i++)
    {
        cout << "\nFlight Number: " << i + 1 << " \n";
        flights[i].get_details();
    }

    return 0;
}
```

OOCP ASSIGNMENT-1

Roll No: 40 **Name: Akshit Trivedi**

Class: MCA-1 **Year: 2021-22**

OUTPUT:

```
C:\MCA\MCA-1\OOCP\Practical Assignment 1\13_Flight.exe

Flight Number: 1
Enter Flight no: 1
Enter Originating Airport code: 123
Enter Destination Airport code: 321
Enter departure time: 12
Enter Arrival time: 15

Flight Number: 2
Enter Flight no: 2
Enter Originating Airport code: 789
Enter Destination Airport code: 987
Enter departure time: 3
Enter Arrival time: 5

Flight Number: 1
Flight no: 1
Originating Airport Code: 123
Destination Airport Code: 321
Departure time: 12
Arrival time: 15

Flight Number: 2
Flight no: 2
Originating Airport Code: 789
Destination Airport Code: 987
Departure time: 3
Arrival time: 5
```