

DAY 2

C#

1.ACCOUNT DETAILS

CODE:

```
using System;

public class Program
{
    public static void Main(string[] args)
    {
        Console.WriteLine("Enter account id");
        int account_id=Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Enter account type");
        String u_account_type=Console.ReadLine();
        Console.WriteLine("Enter account balance");
        int account_baance=Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Enter amount to withdraw");
        int amount_withdraw=Convert.ToInt32(Console.ReadLine());
        Account a=new Account(account_id,u_account_type,account_baance);
        a.GetDetails();
        if(a.WithDraw(amount_withdraw)){

            Console.WriteLine("New Balance:"+a.A_balance);
        }
        else{
            Console.WriteLine("There is no sufficient balance");
        }
    }
}

public class Account{
    int id;
    string accountType;
    double balance;

    public int A_id{
        get{
            return id;
        }
        set{
            id=value;
        }
    }

    public string A_accountType{
        get{
            return accountType;
        }
    }
}
```

```

        set{
            accountType=value;
        }
    }

    public double A_balance{
        get{
            return balance;
        }
        set{
            balance=value;
        }
    }

    public Account(int id,string accountType,double balance){
        this.id=id;
        this.accountType=accountType;
        this.balance=balance;
    }

    public bool Withdraw(double amount){
        if(balance<amount){
            return false;
        }
        else{
            balance=balance-amount;
            return true;
        }
    }

    public string GetDetails(){
        Console.WriteLine("Account Id:"+id);
        Console.WriteLine("Account Type:"+accountType);
        Console.WriteLine("Balance:"+balance);
        return null;
    }
}

```

OUTPUT:

The screenshot shows the Programiz C# Online Compiler interface. The code in Main.cs defines an Account class and a Main method. The Main method prompts the user to enter account ID, type, balance, and amount to withdraw. It then creates an Account object and calls its GetDetails() and Withdraw() methods. The output shows the account details and a successful withdrawal of 2000 units, resulting in a new balance of 3000.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace OnlineCompiler
{
    class Account
    {
        int account_id;
        string account_type;
        int account_balance;
        int account_withdraw;

        public Account(int account_id, string account_type, int account_balance, int account_withdraw)
        {
            this.account_id = account_id;
            this.account_type = account_type;
            this.account_balance = account_balance;
            this.account_withdraw = account_withdraw;
        }

        public void GetDetails()
        {
            Console.WriteLine("Account Id: {0}", account_id);
            Console.WriteLine("Account Type: {0}", account_type);
            Console.WriteLine("Balance: {0}", account_balance);
            Console.WriteLine("There is no sufficient balance");
        }

        public void Withdraw(int amount_withdraw)
        {
            if (account_balance >= amount_withdraw)
            {
                account_balance -= amount_withdraw;
                Console.WriteLine("New Balance: {0}", account_balance);
            }
            else
            {
                Console.WriteLine("There is no sufficient balance");
            }
        }
    }

    class Program
    {
        static void Main()
        {
            Console.WriteLine("Enter account id ");
            int account_id = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter account type");
            string u_account_type = Console.ReadLine();
            Console.WriteLine("Enter account balance");
            int account_balance = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter amount to withdraw");
            int amount_withdraw = Convert.ToInt32(Console.ReadLine());
            Account a = new Account(account_id, u_account_type, account_balance, amount_withdraw);
            a.GetDetails();
            if (a.Withdraw(amount_withdraw))
            {
                Console.WriteLine("New Balance: {0}", a.A_balance);
            }
            else
            {
                Console.WriteLine("There is no sufficient balance");
            }
        }
    }
}
```

Output:

```
mono /tmp/LRLJfxZFY.exe
Enter account id
11
Enter account type
savings
Enter account balance
2000
Enter amount to withdraw
3000
Account Id:11
Account Type:savings
Balance:2000
There is no sufficient balance
=== Code Execution Successful ===
```

The screenshot shows the Programiz C# Online Compiler interface. The code in Main.cs defines an Account class and a Main method. The Main method prompts the user to enter account ID, type, balance, and amount to withdraw. It then creates an Account object and calls its GetDetails() and Withdraw() methods. The output shows the account details and a successful withdrawal of 2000 units, resulting in a new balance of 3000.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace OnlineCompiler
{
    class Account
    {
        int account_id;
        string account_type;
        int account_balance;
        int account_withdraw;

        public Account(int account_id, string account_type, int account_balance, int account_withdraw)
        {
            this.account_id = account_id;
            this.account_type = account_type;
            this.account_balance = account_balance;
            this.account_withdraw = account_withdraw;
        }

        public void GetDetails()
        {
            Console.WriteLine("Account Id: {0}", account_id);
            Console.WriteLine("Account Type: {0}", account_type);
            Console.WriteLine("Balance: {0}", account_balance);
            Console.WriteLine("New Balance: {0}", account_balance);
        }

        public void Withdraw(int amount_withdraw)
        {
            if (account_balance >= amount_withdraw)
            {
                account_balance -= amount_withdraw;
                Console.WriteLine("New Balance: {0}", account_balance);
            }
            else
            {
                Console.WriteLine("There is no sufficient balance");
            }
        }
    }

    class Program
    {
        static void Main()
        {
            Console.WriteLine("Enter account id ");
            int account_id = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter account type");
            string u_account_type = Console.ReadLine();
            Console.WriteLine("Enter account balance");
            int account_balance = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter amount to withdraw");
            int amount_withdraw = Convert.ToInt32(Console.ReadLine());
            Account a = new Account(account_id, u_account_type, account_balance, amount_withdraw);
            a.GetDetails();
            if (a.Withdraw(amount_withdraw))
            {
                Console.WriteLine("New Balance: {0}", a.A_balance);
            }
            else
            {
                Console.WriteLine("There is no sufficient balance");
            }
        }
    }
}
```

Output:

```
mono /tmp/owGYAEbAEN.exe
Enter account id
11
Enter account type
CURRENT
Enter account balance
5000
Enter amount to withdraw
2000
Account Id:11
Account Type:CURRENT
Balance:5000
New Balance:3000
=== Code Execution Successful ===
```

2. CALCULATOR PROGRAM

CODE:

```
using System;

public class Program
{
    public static void Main(string[] args)
    {
        Console.WriteLine("Enter the operator");
        string input=Console.ReadLine();
        Console.WriteLine("Enter the operands");
        int a=Convert.ToInt32(Console.ReadLine());
        int b=Convert.ToInt32(Console.ReadLine());
        Calculator c=new Calculator();
        if(input.Equals("+")){
            Console.WriteLine("Result of "+a+"+"+b+" is "+ c.Addition(a,b));
        }
        else if(input.Equals("-")){
            Console.WriteLine("Result of "+a+"-"+b+" is "+ c.Subtraction(a,b));
        }
        else if(input.Equals("*")){
            Console.WriteLine("Result of "+a+"*"+b+" is "+ c.Multiplication(a,b));
        }
        else if(input.Equals("/")){
            double d;
            Console.WriteLine("Result of "+a+"/"+b+" is "+ c.Division(a,b,out d));
            Console.WriteLine("Remainder: "+d);
        }
        else{
            Console.WriteLine("Invalid Operator");
        }
    }
}

public class Calculator{

    public int Addition(int a,int b){
        return a+b;
    }

    public int Subtraction(int a,int b){
        return a-b;
    }

    public int Multiplication(int a,int b){
```

```

        return a*b;
    }
    public double Division(int a,int b,out double remainder){

        remainder = a%b;
        return a/b;
    }
}

```

OUTPUT:

The screenshot displays the Programiz C# Online Compiler interface. The code editor on the left contains the following C# code:

```

36 - public int Addition(int a,int b){
37     return a+b;
38 }
39
40 }
41 - public int Subtraction(int a,int b){
42     return a-b;
43 }
44 }
45 - public int Multiplication(int a,int b){
46     return a*b;
47 }
48 }
49 - public double Division(int a,int b,out double remainder){
50
51     remainder = a/b;
52     return a/b;
53 }
54 }
55 }

```

The output window on the right shows the execution results:

```

mono /tmp/h9cQubCt6x.exe
Enter the operator
/
Enter the operands
10
9
Result of 10/9 is 1
Remainder: 1
=== Code Execution Successful ===

```

3. Class Program

CODE:

```
using System;

public class Program
{
    public static void Main(string[] args)
    {
        Console.WriteLine("Enter a game");
        String game_name=Console.ReadLine();
        Console.WriteLine("Enter the maximum number of players");
        int no_of_players=Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Enter a game that has time limit");
        String time_limit=Console.ReadLine();
        Console.WriteLine("Enter the maximum number of players");
        int max_of_players_time=Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Enter the time limit in minutes");
        int time_minit=Convert.ToInt32(Console.ReadLine());
        Game g1=new Game(game_name,no_of_players);
        GameWithTimeLimit g=new
GameWithTimeLimit(time_limit,max_of_players_time,time_minit);
        Console.WriteLine(g1);
        Console.WriteLine(g);
    }
}

public class Game{
    public string Name{set; get;}
    public int MaxNumPlayers{set; get;}
    public Game(string Name,int MaxNumPlayers){
        this.Name=Name;
        this.MaxNumPlayers=MaxNumPlayers;
    }
    public override string ToString(){
        return "Maximum number of players for "+Name+" is "+MaxNumPlayers;
    }
}

public class GameWithTimeLimit : Game{
    public int TimeLimit{set; get;}
    public GameWithTimeLimit(string Name,int MaxNumPlayers,int
TimeLimit):base(Name,MaxNumPlayers){

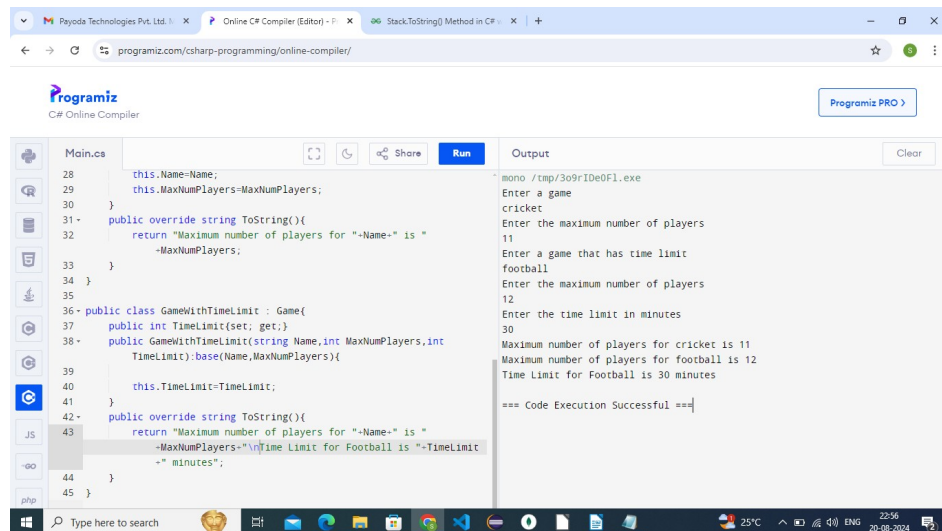
        this.TimeLimit=TimeLimit;
    }
    public override string ToString(){
        return "Maximum number of players for "+Name+" is "+MaxNumPlayers+"\nTime
```

```

Limit for Football is "+TimeLimit+" minutes";
    }
}

```

OUTPUT:



The screenshot shows the Programiz Online C# Compiler interface. The code editor on the left contains the following C# code:

```

Main.cs
28     this.Name=Name;
29     this.MaxNumPlayers=MaxNumPlayers;
30 }
31 public override string ToString(){
32     return "Maximum number of players for "+Name+" is "
33         +MaxNumPlayers;
34 }
35 }
36 public class GameWithTimeLimit : Game{
37     public int TimeLimit{set; get;}
38     public GameWithTimeLimit(string Name,int MaxNumPlayers,int
39         TimeLimit):base(Name,MaxNumPlayers){
40         this.TimeLimit=TimeLimit;
41     }
42     public override string ToString(){
43         return "Maximum number of players for "+Name+" is "
44             +MaxNumPlayers+"\nTime Limit for Football is "+TimeLimit
45             +" minutes";
46     }
47 }

```

The output window on the right shows the execution results:

```

mono /tmp/3o9rIDe0F1.exe
Enter a game
cricket
Enter the maximum number of players
11
Enter a game that has time limit
football
Enter the maximum number of players
12
Enter the time limit in minutes
30
Maximum number of players for cricket is 11
Maximum number of players for football is 12
Time Limit for Football is 30 minutes

=== Code Execution Successful ===

```