IARE

IARE
INSTITUTE OF
AERONAUTICAL ENGINEERING
(An Autonomous Institute affiliated to JNTUH, Hyderabad)
Dundigal, Hyderabad - 500 043

LABORATORY WORK BOOK

Name of the Student : Mahek Tabeen

Class : CSE – C Semester : III

Course Code : AITD02 Course Name : PWO Lab

Name of the Course Faculty : Mr. P Suresh Kumar Faculty ID : IARE 11068

Exercise Number : 2 Week Number : 2 Date : 21/9/24

Roll Number									
2	3	9	5	1	A	0	5	4	U

S. No.	Exercise Number	EXERCISE NAME	MARKS AWARDED						
			Aim/ Preparation	Algorithm / Procedure / Performance in the Lab		Source Code / Calculations and Graphs	Program Execution / Results and Error Analysis	Viva - Voce	Total
			4	4		4	4	4	20
1	2.1	Arrow Pattern.	4	2	2	4	4	4	20
2	2.2	Working 9 to 5							
3	2.3	New Driving license							
4	2.4	Who won the Leagne.							
5	2.5	The Day Robery Born in Dutch							
6	2.6	Smallest missing +ve integer.	4	2	2	4	4	4	20
7	2.7	Brief case Lock.							
8	2.8	Keeping Count							
9	2.9	Break Point							
10	2.10	ATM Transactions.							
11									
12									

Mahek
Signature of the Student

Signature of the Faculty

START WRITING FROM HERE.

2.1 → Arrow Pattern

```
using System;
using System.Collections.Generic;
class Program
{
    static void Main()
    {
        int num = 4;
        List<string> arrowPattern = Arrow(num);
        Console.WriteLine("[" + string.Join(",", arrowPattern) + "]");
    }
    static List<string> Arrow(int num)
    {
        List<string> pattern = new List<string>();
        for(int i = 1; i <= num; i++)
        {
            pattern.Add(new string('>', i));
        }
        for(int i = num; i > 0; i--)
        {
            Pattern.Add(new string('>', i));
        }
        return pattern;
    }
}
```

Output:

[>,>>,>>>,>>>>,>>>,>>,>]

ROLL NUMBER :

2.2 Working 9 to 5

```
Using System;
Public class Challenge.
{
    Public static void main(string[] args)
    {
        Console.WriteLine(OverTime(new double[] {9,17,30,1.5}));
        Console.WriteLine(OverTime(new double[] {16,18,30,1.8}));
        Console.WriteLine(OverTime(new double[] {13.25,15,30,1.5}));
    }
    Public static string OverTime(double[] arr)
    {
        double start = arr[0];
        double finish = arr[1];
        double rate = arr[2];
        double mult = arr[3];
        double pay = 0;
        if (start < 17)
        {
            Pay += (Math.Min(17, finish) - start) * rate;
        }
        if (finish > 17)
        {
            Pay += (finish - Math.Max(17, start)) * rate * mult;
        }
        return string.Format("${0:0.00}", pay);
    }
}
```

Output:

$ 240.00
$ 84.00
$ 52.50

ROLL NUMBER :

2.3 New Driving License

```
using System;
Public class LinearProcessing.
{
  Public Static int License (string me, int agents. string others)
  {
    String [] names = (me + " " + others). Split (' ');
    Array.Sort (names);
    int position = Array. IndexOf (names, me)+1;
    int timeTo process = ((position-1) /agents)* 20+20;
    return timetoprocess;
  }
   Public Static Void Main (String [] args)
  {
   console. WriteLine (License ("Eric", 2, "adam caroline Rebecca Frank"));
   Conside. writeLine (License ("zebedian", 1, "bob jn becky pat "));
   console.writeLine (License ("Aaron", 3," jane maxolivia sam "));
  }
}
```

Output

40

100

20.

ROLL NUMBER :

2.4 Who won the league ?

```
using System;
using System.Collections.Generic;
using System.Linq;
Public class EPL
{
    Public static string EPLResult(string[] table, string teamName)
    {
        List<(string Team, int Points, int GoalDifference)> team = new
            List<(string, int, int)>();
        foreach (var entry in table)
        {
            var parts = entry.Split(',');
            string team = parts[0].Trim();
            int won = int.Parse(parts[3].Trim());
            int goalDifference = int.Parse(parts[5].Trim());
            int points = won * 3 + drawn;
            teams.Add((team, points, goalDifference));
        }
        var sortedTeams = teams.OrderByDesce(t => t.Points).
            ThenByDescen(t => t.GoalDifference).ToList();
        int position = sortedTeams.FindIndex(t => t.Team == teamName)+1;
        var teamData = sorted Teams.First(t => t.Team == teamName);
        string ordinalPosition = position + (position == 1 ? "st":
            Position == 2 ? "nd": position == 3 ? "rd" : "th");
        return $"{teamName} came {ordinalPosition} with {teamData.
            Points};
    }
    PSVM {
        string[] table = {
```

```
        "Arsenal, 38, 14, 14, 10, 8",
        "Aston Villa, 38, 9, 8, 21, -26",
        "Bournemouth, 38, 9, 7, 22, -26",
        "Brighton, 38, 9, 14, 15, -15",
        "Burnley, 38, 15, 9, 14, -7",
        "Chelsea, 38, 20, 6, 12, 15",
        "Coystal palace, 38, 11, 10, 17, -19",
        "Everton, 38, 13, 10, 15, -12",
        "West Ham, 38, 10, 9, 19, -13",
        "Wolves, 38, 15, 14, 9, 11"
    };
    Console.WriteLine(EPLResult(table, "Liverpool"));
    Console.WriteLine(EPLResult(table, "Manchester Utd"));
    Console.WriteLine(EPLResult(table, "Norwich"));
  }
}
```

Output:

Liverpool came 1st with 99 points and a goal diff of 52!
Manchester Utd came 3rd with 66 points and goal diff of 30!
Norwich came 20th with 21 points and a goal diff of -49!

2.5→ The Day Rob Was Born in Dutch.

```
using System;
using System.Globalization;
public class WeekdayInDutch
{
    public static string WeekdayRobWasBornInDutch(int year,
        int month, int day)
    {
```

```
    DateTime = new DateTime(year, month, day);
    CultureInfo dutchCulture = new CultureInfo("nl-NL");
    string weekdayInDutch = date.ToString("dddd", dutchCulture);
    return weekdayInDutch;
}
Public static void Main(string[] args)
{
    Console.WriteLine(WeekdayRobWasBornInDutch(1970, 9, 21));
    Console.WriteLine(WeekdayRobWasBornInDutch(1945, 9, 2));
    Console.WriteLine(WeekdayRobWasBornInDutch(2001, 9, 11));
}
}
```

Output:

```
maandag
zondag
dinsdag
```

2.6 Smallest Missing Positive Integer

```
using System;
using System.Linq;
Public class SmallestMissingPositive
{
    Public static int MinMissPos(int[] arr)
    {
        Array.Sort(arr);
        int SmallestMissingPositive = 1;
        foreach(int num in arr)
        {
            if(num <= 0)
            continue;
            if(num = SmallestMissingPositive)
            {
```

```
            SmallestMissingPositive++;
        }
    }
    return SmallestMissingPositive;
}
Public static void Main(string[] args)
{
    int[] arr1 = {-2,6,4,5,7,-1,1,3,6,-2,9,10,2,2};
    int[] arr2 = {5,9,-2,0,1,3,9,3,8,9};
    int[] arr3 = {0,4,4,-1,9,4,5,2,10,7,6,3,10,9};
    Console.WriteLine(MinMissPos(arr1));
    Console.WriteLine(MinMissPos(arr2));
    Console.WriteLine(MinMissPos(arr3));
    }
}
```

Output:

```
8
2
1
```

2.7 Briefcase Lock.

```
using System;
Public class LockTurns
{
    public static int MinTurns(string current, string target)
    {
        int totalTurns = 0;
        for(int i = 0; i < current.Length; i++)
        {
            int currentDigit = int.Parse(current[i].ToString());
            int targetDigit = int.Parse(target[i].ToString());
            int forwardTurns = (targetDigit - currentDigit + 10)%10;
```

```
        int backwardTurns = (currentDigit - targetDigit + 10) % 10;
        totalTurns += Math.Min(forwardTurns, backwardTurns);
    }
    return totalTurns;
}
Public static void Main(string[] args)
{
    Console.WriteLine(MinTurns("4089", "5672"));
    Console.WriteLine(MinTurns("1111", "1100"));
    Console.WriteLine(MinTurns("2391", "4984"));
}
}
```

Output:

9
2
10

2.8 Keeping Count

```
using System;
using System.Collections.Generic;
using System.Linq;
Public class DigitCounter
{
    Public static int DigitCount(int number)
    {
        string numberStr = number.ToString();
        Dictionary<char, int> digitCounts = new Dictionary<char,
                                            int>();
        foreach (char digit in numberStr)
        {   if (digitCounts.ContainsKey(digit))
```

```
        {
            digitCounts[digit]++;
        }
        else {
            digitCounts[digit] = 1;
        }
    }
    string resultStr = "";
    foreach (char digit in numberStr)
    {
        resultStr += digitCounts[digit];
    }
    return int.Parse(resultStr);
}
Public static void Main(string[] args)
{
    Console.WriteLine(DigitCount(221333));
    Console.WriteLine(DigitCount(136116));
    Console.WriteLine(DigitCount(2));
}
}
```

Output:

221333

312332

1.

ROLL NUMBER :

2.9 Break Point.

```
using System;
Public class BreakPointChecker
{
  Public static bool BreakPoint(int number)
  {
     string numberStr = number.ToString();
     int length = numberStr.Length;
     for (int j = 0; j < i ; j++)
     {
        leftSum += int.Parse(numberStr[j].ToString());
     }
     int rightSum = 0;
     for (int j = i ; j < length ; j++)
     {
        rightSum += int.Parse(numberStr[j].ToString());
     }
     if (leftSum = rightSum)
     {
        return true;
     }
  }
  return false;
}
Public static void Main(string[] args)
{
    Console.WriteLine(BreakPoint(159780));
    Console.WriteLine(BreakPoint(112));
    Console.WriteLine(BreakPoint(1034));
    Console.WriteLine(BreakPoint(10));
    Console.WriteLine(BreakPoint(343));
  }
}
```

Output:

True
True
True
False
False

2.10 ATM Transactions:

```
using System;
Public class ATM
{
    Private static int pin = 4040;
    Private static int balance = 10000;
    Public static void Main(String[] args)
    {
        Console.WriteLine("Welcome to the ATM");
        bool isAuthenticated = AuthenticateUser();
        if (isAuthenticated)
        {
            ShowMenu();
        }
        else
        {
            Console.WriteLine("Invalid PIN. Access denied.");
        }
    }
Private static bool AuthenticateUser()
{
    Console.Write("Enter the PIN: ");
    int enteredPin;
```

```
        bool isValidPin = int.TryParse(Console.ReadLine(), out
                                enteredPin);
        if (isValidPin && enteredPin = pin)
        {
            return true;
        }
        return false;
    }
    Private static void ShowMenu()
    {
        bool continueRunning = true;
        while (continueRunning)
        {
            Console.WriteLine("\n1. To check balance");
            Console.WriteLine("2. To withdraw money");
            Console.WriteLine("3. To deposit money");
            Console.WriteLine("4. To change the PIN");
            Console.WriteLine("5. To exit");
            Console.Write("Enter your choice: ");
        int choice;
        bool isValidChoice = int.TryParse(Console.ReadLine(),
                                out choice);
        if (!isValidChoice || choice < 1 || choice > 5)
        {
            console.WriteLine("Invalid choice. Please try
                                again.");
            Continue;
        }
        Switch (choice)
        {
```

```
            case 1:
            CheckBalance();
            break;
            case 2:
            WithdrawMoney();
            break;
            case 3:
            DepositMoney();
            break;
            case 4:
            ChangePIN();
            break;
            case 5:
            continueRunning = false;
            Console.WriteLine("Thank you for using the ATM.");
            break;
        }
    }
}
Private static void CheckBalance()
{
    Console.WriteLine($"The current balance in your account
                is {balance}");
}
Private static void CheckBalance WithdrawMoney()
{
    Console.Write("Enter the amount to withdraw: ");
    int amount;
    bool isValidAmount = int.TryParse(Console.ReadLine(),
                                      out amount);
```

```
        if (!isValidAmount || amount <= 0 || amount % 100 != 0)
        {
            Console.WriteLine("Invalid amount. Please enter a
                                multiple of 100.");
            return;
        }
        if (amount > balance)
        {
            Console.WriteLine("Insufficient balance.");
            return;
        }
        balance -= amount;
        Console.WriteLine($"Please collect the cash: {amount}");
        Console.WriteLine($"The current balance is now: {balance}");
    }
    Private static void DepositMoney()
    {
        Console.Write("Enter the amount to deposit: ");
        int amount;
        bool isValidAmount = int.TryParse(Console.ReadLine(), out
                                amount);
        if (!isValidAmount || amount <= 0)
        {
            Console.WriteLine("Invalid amount. Please enter a +ve
                                number.");
            return;
        }
        balance += amount;
        Console.WriteLine($"The current balance is now: {balance}");
    }
    Private static void ChangePIN()
    {
```

```
Console.Write("Enter the current PIN: ");
int currentPin;
bool isValidPin = int.TryParse(Console.ReadLine(), out);
if (currentPin != Pin)
{
    Console.WriteLine("Incorrect PIN.");
    return;
}
Console.Write("Enter the new PIN: ");
int newPin;
bool isValidNewPin = int.TryParse(Console.ReadLine(), out);
if (newPin < 1000 || newPin > 9999)
{
    Console.WriteLine("Invalid PIN. PIN must be a
    4-digit number.");
    return;
}
Pin = newPin;
Console.WriteLine("PIN successfully changed.");
    }
}
```

Output:

```
Welcome to the ATM
Enter the PIN: 4040
1) To check balance
2) To withdraw money
3) To deposit money
4) To change PIN
5) To Exit.
Enter your choice: 1
The current balance in your account is 10000.
```

Kanika