דף עבודה מערכים 2 – אופיר הופמן י3

תרגיל 11

```
int n = int.Parse(Console.ReadLine());

char[] arr = new char[n];

for (int i = 0; i < arr.Length; i++)
{
    arr[i] = char.Parse(Console.ReadLine());
}

for (int i = 0; i < arr.Length; i++)
{
    char letter = arr[i];
    bool found = false;
    for (int index = 0; index < arr.Length && !found; index++)
    {
        if (arr[index] == letter+1)
        {
            found = true;
            Console.WriteLine(letter + "found");
        }

    }
}
```

תרגיל 12

```
public static int TavCount(char[] arr, char ch)
{
    int count = 0;
    for (int i = 0; i < arr.Length; i++)
    {
        if (arr[i] == ch)
            count++;
    }
    return count;
}

public static void Ex12(char[] arr)
{
    bool found = false;
    int indexStart = -1;

    for (int i = 0; i < arr.Length && !found; i++)
    {
        if (TavCount(arr, arr[i]) >= 3)
        {
            found = true;
            indexStart = i;
        }
    }

    if (found)
        Console.WriteLine(indexStart);
    else
        Console.WriteLine("Didnt find");

}
```

תרגיל 13

```
public static bool IsIn(int[] arr, int num)
{
    bool found = false;
    for (int i = 0; i < arr.Length && !found; i++)
    {
        if (arr[i] == num)
            found = true;
    }
    return found;
}

public static bool Ex13(int[] arr)
{
    bool cont = true;
    for (int i = 1; i <= arr.Length; i++)
    {
        if (IsIn(arr, i) == false)
            cont = false;
    }
    return cont;
}
```

תרגיל 14

```
//EX14
public static void MoveArr(int[] arr, int firstIndex, int lastIndex)
{
    for (int i = lastIndex; i >= firstIndex; i--)
    {
        arr[i + 1] = arr[i];
    }
}

public static void Ex14()
{
    Console.WriteLine("Enter number:");
    int n = int.Parse(Console.ReadLine());

    int[] arr = new int[n];

    Console.WriteLine("Enter value:");
    int num = int.Parse(Console.ReadLine());
    arr[0] = num;

    int lastNumIndex = 0;

    for (int i = 1; i < arr.Length; i++)
    {
        Console.WriteLine("Enter value:");
        num = int.Parse(Console.ReadLine());

        bool found = false;
        int insertIndex = 0;
        for (int index = 0; index < arr.Length && !found; index++)
        {
```

```
            if (arr[index] > num)
            {
                found = true;
                insertIndex = index;
            }
        }

        if (found)
        {
            MoveArr(arr, insertIndex, lastNumIndex);
            arr[insertIndex] = num;
            lastNumIndex++;
        }
        else
        {
            arr[lastNumIndex + 1] = num;
            lastNumIndex++;
        }

    }

    PrintArr(arr);
}
```

תרגיל 15

```
public static void MoveArr1(int[] arr)
{
    for (int i = arr.Length-1; i >= 0; i--)
    {
        arr[i + 1] = arr[i];
    }
}
```

המשך למטה

תרגיל 16

```
public static void Ex16()
{
    Console.WriteLine("Enter Show number (1-15):");
    int show = int.Parse(Console.ReadLine());

    Console.WriteLine("Enter number of tickets:");
    int tickets = int.Parse(Console.ReadLine());

    int sum = 0;
    int[] arr = new int[16];

    while (show != 0 && tickets != 0)
    {
        sum += tickets;
        arr[show] += tickets;

        Console.WriteLine("Enter Show number (1-15):");
        show = int.Parse(Console.ReadLine());

        Console.WriteLine("Enter number of tickets:");
        tickets = int.Parse(Console.ReadLine());
    }

    for (int i = 1; i < arr.Length; i++)
    {
        double precentage = (double)(((double)arr[i] / sum) * 100);
        Console.WriteLine("Show " + i + ": " + (precentage)+"%");
    }
}
```

תרגיל 17

```
public static void Ex17()
{
    int[] N = new int[10];
    Random rnd = new Random();
    for(int i = 0; i < N.Length; i++)
    {
        N[i] = rnd.Next(1, 10);
    }

    int oddIndex = 0;
    int evenIndex = N.Length - 1;
    int[] M = new int[10];

    for (int i = 0; i < N.Length; i++)
    {
        if (N[i] % 2 == 0)
        {
            M[evenIndex] = N[i];
            evenIndex--;
        }
        else
        {
            M[oddIndex] = N[i];
            oddIndex++;
        }
    }
}
```

תרגיל 18

```
public static int MaxIndex(double[] arr)
{
    int maxIndex = 0;
    for (int i = 0; i < arr.Length; i++)
    {
        if (arr[i] > arr[maxIndex])
            maxIndex = i;
    }
    return maxIndex;
}
public static void Ex18()
{
    double[] arr = new double[6];
    for (int i = 0; i < arr.Length; i++)
    {
        arr[i] = double.Parse(Console.ReadLine());
    }

    int[] newArr = new int[arr.Length];

    for (int i = 0; i < arr.Length; i++)
    {
        newArr[i] = MaxIndex(arr) + 1;
        arr[MaxIndex(arr)] = 0;
    }

    PrintArr(newArr);
}
```

תרגיל 19 למטה

תרגיל 19

```
public static void Ex19()
{
    int[] arr = new int[20];
    Random rnd = new Random();
    for (int i = 0; i < arr.Length; i++)
    {
        arr[i] = rnd.Next(-20, 21);
    }
    PrintArr(arr);
    int[] newArr = new int[arr.Length + 1];
    int sum = 0;
    for (int i = 0; i < newArr.Length; i++)
    {
        newArr[i] = sum;
        if (i < arr.Length)
            sum += arr[i];
    }
    Console.WriteLine();
    PrintArr(newArr);
}
```

תרגיל 20 למטה

תרגיל 20

א. אינדקס ראשון שגדול ב1 מx ועוד אינדקס שקטן ב1 מx, כל פעם מגדילים כל אחד מהם ובודקים האם אחד מהם פנוי (יעיל יותר):

```
public static int Ex20A(string[] arr, int x)
{
    if (arr[x] == "")
        return x;

    int rightIndex = x+1, leftIndex = x-1;
    bool found = false;

    for (int i = 0; i < arr.Length/2 && !found; i++)
    {
        if (arr[rightIndex] == "")
            return leftIndex;
        else if (arr[leftIndex] == "")
            return leftIndex;

        rightIndex++;
        leftIndex--;
    }
    return -1;
}
```

ב. לעבור על המערך מההתחלה ועד הסוף ולבדוק מה המושב הפנוי הקרוב ביותר לx (פחות יעיל):

```
public static int Ex20B(string[] arr, int x)
{
    if (arr[x] == "")
        return x;

    int closest = int.MaxValue;
    for (int i = 0; i < arr.Length; i++)
    {
        if (arr[i] == "" && Math.Abs(x-i) < closest)
        {
            closest = i;
        }
    }
    return closest;
}
```