BSP 581 BİLGİSAYAR PROGRAMLAMA LABORATUVARI
5.1 HAFTA LABORATUVAR DÖKÜMANI

Exercise 1 - Başlangıç ve bitiş değerleri alan bu değerler arasında random integer sayı döndüren fonksiyonu yazın. Fonksiyonun prototipi aşağıdaki gibidir.

public static int random(start, end)

Exercise 2 - Başlangıç ve bitiş değeri alan ve bu değerler arasında count değeri kadar random sayı döndüren fonksiyonu yazın. Fonksiyonun prototipi aşağıdaki gibidir.

Public static int[] createRandomList(start, end, count)

Exercise 3 - Parametre olarak sayı alan ve bu sayının ikinin kuvveti olup olmadığını kontrol eden fonksiyonu yazınız. Fonksiyonun prototipi aşağıdaki gibidir.

Public static boolean isPowerOfTwo(number)

Exercise 4 - Kendisine gönderilen saat ve dakika değerlerini kullanarak akrep ile yelkovan arasındaki açıyı, "derece" cinsinden hesaplayarak geri dondüren "getAngle" isimli fonsiyonu yazın. Fonksiyonun prototipi aşağıdaki gibidir.

public static double getAngle(int hour, int minute)

Exercise 5 - x, y pozitif tamsayılar olmak üzere, eğer x sayısının çarpanları toplamı y sayısına, ve aynı zamanda y sayısının çarpanları toplamı x sayısına eşit ise, bu sayılar "arkadaştır" denir. Örneğin 220 ve 284 arkadaş sayılardır:

```
220 => 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284
284 => 1 + 2 + 4 + 71 + 142 = 220
```

Kendisine gönderilen iki tamsayının arkadaş olup olmadıklarını sınayan, areFriends fonksiyonunu tanımlayın:

```
Public static boolean areFriends(int number1, int number2)
```

Exercise 6 - Verilen sayının asal olup olmadığını kontrol eden fonksiyonu yazınız. Fonksiyonun prototipi aşağıdaki gibidir.

Public static boolean isPrime(number)

Exercise 7 - Başlangıç ve bitiş değeri verilen aralıkta bulunan tüm asal sayıları bulan fonksiyonu yazınız. Fonksiyon içinde asal sayı kontrolü için isPrime(number) fonksiyonunu kullanınız. Fonksiyonun prototipi aşağıdaki gibidir.

Public static int[] findAllPrimeNumbers(start, end)

Exercise 8 - Kendisine gönderilen int türden bir sayının basamak sayısına geri dönen numberOfDigits fonksiyonunu tanımlayın. Fonksiyonun prototipi aşağıdaki gibidir.

```
Public static int numberOfDigits(int value)
```

Exercise 9 - Parametre olarak tam sayı alan ve bu tamsayının basamak değerlerini toplamını döndüren fonksiyonu yazınız. Fonksiyonun prototipi aşağıdaki gibidir.
Public static int sumDigits(number)

Exercise 10 - Kendisine gönderilen Celsius degerinin Fahrenheit eşdeğeri ile geri dönen `cel_to_fahr` isimli fonksiyonu yazın. F = C*1.8000 4 32. Fonksiyonun prototipi aşağıdaki gibidir.

```
Public static int celsiusToFahrenheit(int celsius)
```

Exercise 11 - Kendisine gönderilen pozitif bir tamsayının kendisi hariç tüm çarpanlarının toplamı ile geri dönen `sumFactors` fonksiyonunu tanımlayınız. Fonksiyonun prototipi aşağıdaki gibidir.

```
Public static int sumFactors(int value);
```

Exercise 12 - int türden bir dizinin küçükten büyüğe göre sıralı olup olmadığını sınayan isSorted isimli fonksiyonu yazınız. Fonksiyonun prototipi aşağıdaki gibidir.

```
Public static isSorted(int[] array);
```

Exercise 13 - int türden bir dizinin en büyük elemanını bulan bir kod parçası yazınız. Fonksiyonun prototipi aşağıdaki gibidir. Kodunuzu test etmek için createRandomList(start, end, count) fonksiyonun kullanarak 0 ile 100.000 arasında sayılardan oluşan 1.000 elamanlı bir liste oluşturun ve maxItemInArray fonksiyonu kullanarak bu liste içerisindeki en büyük elamanı bulun.

Public static int maxItemInArray(int[] array)

Exercise 14 – int türden bir dizinin en büyük ikinci elemanını bulan bir kod parçası yazınız. Fonksiyonun prototipi aşağıdaki gibidir. Kodunuzu test etmek için createRandomList(start, end, count) fonksiyonun kullanarak 0 ile 100.000 arasında sayılardan oluşan 1.000 elamanlı bir liste oluşturun ve maxSecondItemInArray fonksiyonu kullanarak bu liste içerisindeki en büyük elamanı bulun.

Public static int maxSecondItemInArray(int[] array)

Exercise 15 – int türünden bir dizi içerisindeki en çok yinelenen sayıyı bulunuz. En çok yinelenen sayı birden fazla ise dizi içerisinde ilk görüleni bulunacak. Kodunuzu test etmek için createRandomList(start, end, count) fonksiyonun kullanarak 0 ile 100 arasında sayılardan oluşan 100 elamanlı bir liste oluşturun ve fonksiyonu kullanarak bu liste içerisindeki en fazla yinelenen elamanı bulun.

Public static int findMaximumRecurrenceNumber(int[] array)

Exercise 16 – Verilen sayının faktoriyeli bulan fonksiyonun yazın. Fonksiyonun prototipi aşağıdaki gibidir.

Public static int faktöriyel(number)

Exercise 17 – e sayısını aşağıdaki serinin toplamı şeklinde hesaplayan bir program yazın. Faktoriyel fonksiyonunu kullanınız. e = 1/0! + 1/1! + 1/2! + 1/3! + 1/4! + 1/5! ……..+ 1/n!

Public static float eNumbervalue(n)

Exercise 18 – Standart sapma ortalamadan sapmanın bir ölçüsüdür. Standart sapmayı aşağıdaki formül ile hesaplayabilirsiniz:

```
ortalama       = dizinin aritmetik ortalaması
n              = dizinin eleman sayısı
a[i]           = dizinin her bir elemanı
farkkare[i]    = (a[i] - ortalama) * (a[i] - ortalama)
standart sapma = karekok (kumulatif(farkkare[i]) / n)
```

Yukarıdaki formülde ortalama dizinin aritmetik ortalamasını n ise dizinin eleman sayısını göstermektedir. Bir dizinin standart sapması dizi elemanlarının ortalamadan farklarının kareleri kümülatif toplamının dizinin eleman sayısına bölümünün karekökünе eşittir.

Public static int findStandartDevation(int[] array)

Kodunuzu test etmek için createRandomList(start, end, count) fonksiyonun kullanarak 0 ile 100 arasında sayılardan oluşan 10 elamanlı bir liste oluşturun ve findStandartDeviation fonksiyonu kullanarak bu listenin standart sapmasını hesaplayın.