Lösningsförslag: Instuderingsfrågor, del D

Uppgift 1.

a) Inget fält behövs. Man kan läsa in ett tal i taget och addera dessa till summan.
b) Här behövs ett fält.
c) Här behövs inget fält. Talen kan läsas ett i taget och hela tiden sparar man undan det hittills största talet.
d) Här behövs ett fält.
e) Här behövs inget fält. För att beräkna medelvärdet räcker att beräkna summan av talen.
f) Här behövs ett fält.
g) Här behövs ett fält
h) Här behövs ett fält.

Uppgift 2.

Fältet rymmer 4 element. Elementen i fältet är av datatypen **int**. Fältet är indexerat från 0 till 3.

Uppgift 3.

a) **char** vokaler = **new** char[5]; — **char** i högerledet och **char**[5] i vänsterledet
b) **double**[] m = **double**[10]; — **new** saknas i högerledet
c) **int**[]odd = **new int**{1, 3, 5, 7 , 9}; — { och } istället för [och] i vänsterledet

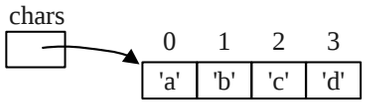
Uppgift 4.

a) **int**[] talen = {3, 19, 7, 9, 4};
b) **int**[] talen = {'0' , '1' , '2' , '3' , '4' , '5', '6' , '7' , '8' , '9' };
c) Point[] punkter = **new** Point[25];

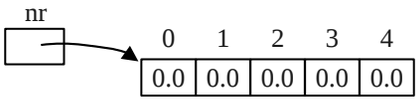
Uppgift 5.

a)

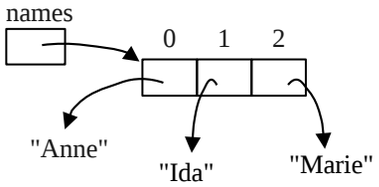
b)



c)



d)



Uppgift 6.

Utskrifterna blir:

a) Olika
b) Lika

Orsaken till att utskriften i deluppgift a) blir "Olika" beror på att ett fält är ett objekt och att därför är en fältvariabel är en referensvariabel. Således är det referenserna fälten som jämförs och inte innehållet i de båda fälten. För att jämföra innehållet i fälten kan, som i deluppgift b), klassmetoden equals i klassen Arrays användas.

Uppgift 7.

a) 25 b) 13 c) 12 d) 6 e) 1

Uppgift 8.

a) [1, 3, 5, 7, 9, 9, 9, 9, 9]
b) [2, 3, 4, 5, 4, 3, 2, 1, 1]
c) [1, 1, 1, 1, 1, 1, 1, 1, 1]
d) [1, 1, 1, 1, 1, -1, -1, -1, -1]
e) [5, 4, 3, 2, 5, 4, 3, 2, 1]

Uppgift 9.

a) 3
b) Vi får ett exekveringsfel! Detta beroende på att talen i serien är sorterade i växande ordning. Således kommer **while**-satsen att löpa i genom samtliga element i fältet. När de två siste elementen jämförs, dvs elementen med värdena 49 respektive 53, evalueras villkoret i **while**-satsen till **true** och **while**-satsen genomlöps ytterligare en gång. Nu inträffar emellertid följande: variabeln i har värdet 5 och villkoret i **while**-satsen innebär alltså att fältelementen vekt[5] och vekt[6] (skall jämföras. Men det finns inget fältelement vekt[6]!!!

Uppgift 10.

[1, -1, -2, -1, 1, 2, 1, -1, -2]

Uppgift 11.

a) **for**-satsen kommer att genomlöpas då k har värdet 10, men fältet values har ingen element med index 10. Ett exekveringsfel kommer att inträffa. **for**-satsen bör ha följande utseende:

```
for (int k = 0; k < values.length; k = k + 1)
```

b) Det finns inget fält att genomlöpa. Deklarationen skapar en variabel values som kan referera till ett heltalsfält, men inget fält har skapas varför values refererar till **null**.

c) **for**-satsen kommer att genomlöpas då k har värdet 7. Inuti **for**-satsen finns en refereras till elementet values[k + 1], men fältet values har ingen element med index 8. Ett exekveringsfel kommer att inträffa. **for**-satsen bör ha följande utseende:

```
for (int k = 0; k < values.length - 1; k = k + 1)
```

Uppgift 12.

Utskriften blir:

7

Uppgift 13.

[1, 1, 2, 3, 5, 8, 13]

Uppgift 14.

[1, 5, 6, 9]

Uppgift 15.

Utskriften blir:

```
x= 5
v[0] = 3
v[1] = 2
v[2] = 3
```

Förklaring:

Värdet av de aktuella parametrarna kopieras in i de formella parametrarna, vilket betyder att en aktuell parameter som är en enkel datatyp kommer att ha samma värde efter metodanropet som innan, dvs x kommer att ha kvar värdet 5. Är den aktuella parametern ett objekt innebär det att den aktuella parametern och den formella parametern refererar till samma objekt, och att de förändringar som utförs i metoden på objektet som refereras av den formella parametern också utförs på objektet som refereras av den aktuella parametern (är ju ett och samma objekt).

Uppgift 16.

```
111
2
666
```

Uppgift 17.

```
for (int i = tal.length - 1; i >= 0; i = i - 1)
    System.out.println(tal[i]);
```

Uppgift 18.

Enklast är nog att använda sig av två **for**-satser enligt följande:

```
for (int i = 0; i <= vekt.length; i = i + 2)
    vekt[i] = 1;
for (int i = 1; i <= vekt.length; i = i + 2)
    vekt[i] = 0;
```

Alternativt kan man använda en **for**-sats och en **if**-sats enligt nedan:

```
for (int i = 0; i <= vekt.length; i = i + 1) {
    if (i%2 == 0)
        vekt[i] = 1;
    else
        vekt[i] = 0;
}
```

Uppgift 19.

Förvillkoret är

```
//before: vekt är inte null
```

Om vekt är **null** kommer ett exekveringsfel att inträffa när vekt.length refereras.

Uppgift 20.

Förvillkoren är

```
//before: vekt är inte null och vekt innehåller minst ett element
```

Om vekt är **null** kommer ett exekveringsfel att inträffa när vekt.length refereras och om vekt innehåller 0 element kommer en division med 0 att inträffa i uttrycket sum/vekt.length.

Uppgift 21.

```
public static int countOdd(int[] vekt) {
    int nrOfOdd = 0;
    for (int i = 0; i < vekt.length; i = i + 1) {
        if (vekt[i] % 2 == 1) {
            nrOfOdd = nrOfOdd + 1;
        }
    }
    return nrOfOdd;
}// countOdd
```

Uppgift 22.

```
public static int computeOddSum(int[] vekt) {
    int sumOfOdd = 0;
    for (int i = 0; i < vekt.length; i = i + 1) {
        if (vekt[i] % 2 == 1) {
            sumOfOdd = sumOfOdd + vekt[i];
        }
    }
    return sumOfOdd;
}//computeOddSum
```

Uppgift 23.

```
public static int countCloseTo(double[] values, double target, double tolerance) {
    int count = 0;
    for (int i = 0; i < values.length; i = i + 1) {
        if ( Math.abs(target - values[i]) < tolerance) {
            count = count + 1;
        }
    }
    return count;
}// countCloseTo
```

Uppgift 24.

```
public static double[] doubleValues(double[] original) {
    double[] doubled = new double[original.length];
    for (int i = 0; i < original.length; i = i + 1) {
        doubled[i] = original[i] * 2;
    }
     return doubled;
}// doubleValues
```

Uppgift 25.

```
public static boolean[] reverse(boolean[] original) {
    boolean[] reversed = new boolean[original.length];
     for (int i=0 ; i < reversed.length; i = i + 1) {
        reversed[i] = original[original.length - i - 1];
    }
     return reversed;
}// reverse
```

Uppgift 26.

```
public static double[] removeFirst(double[] original) {
    double[] shorter = new double[original.length - 1];
    for (int i = 1; i < original.length; i = i + 1) {
        shorter[i - 1] = original[i];
    }
    return shorter;
}// removeFirst
```

Uppgift 27.

```
public static int[] removeAt(int[] original, int index) {
    int[] shorter = new int[original.length - 1];
    for (int i = 0; i < index; i = i + 1) {
        shorter[i] = original[i];
    }
    for (int i = index+1; i < original.length; i = i + 1) {
        shorter[i - 1] = original[i];
    }
    return shorter;
}//removeAt
```

Uppgift 28.

```
public static int findFirstTarget(int[] values, int target) {
    for (int i=0; i < values.length; i = i + 1) {
        if (values[i] == target) {
            return i;
        }
    }
    return -1;
}// findFirstTarget
```

Uppgift 29.

```
public static int[] removeAllTargets(int[] original, int target) {
    int location = findFirstTarget(original, target);
    while (location >= 0 ) {
        original = removeAt(original, location);
        location = findFirstTarget(original, target);
    }
    return original;
}//removeAllTargets
```

Uppgift 30.

a) 27	b) h	c) 2
d) 10	e) three	f) THE THREE DID FEED THE DEER
g) -1	h) 25	i) Tha thraa did faad tha daar

Uppgift 31.

a) ett tal större än 0	b) ett tal mindre än 0	c) 0

Uppgift 32.

a) **false**	b) **false**	c) **true**
d) **true**	e) **false**	f) **true**

Uppgift 33.

Utskriften blir :

snusfabrik
fabrikör

Uppgift 34.

Utskriften blir:

optr

Uppgift 35.

Utskriften blir:

utercom

Uppgift 36.

Lämpligast är att införa en metod:

```
String str = "Detta ÄR en STRÄNG med BÅDE små OcH STORA bOkStÄvEr!";
str = exchangeCapitalAndLower(str);
```

Metoden får följande utseende:

```
public static String exchangeCapitalAndLower(String str) {
    String strCopy = "";
    for (int i = 0; i < str.length(); i = i + 1) {
        if ( Character.isLowerCase(str.charAt(i)))
            strCopy = strCopy + Character.toUpperCase(str.charAt(i));
        else if ( Character.isUpperCase(str.charAt(i)))
            strCopy = strCopy + Character.toLowerCase(str.charAt(i));
        else
            strCopy = strCopy + str.charAt(i);
    }
    return strCopy;
}// exchangeCapitalAndLower
```

Uppgift 37.

Utskriften blir:

```
world  6
world  6
```

Uppgift 38.

```
public static String fixEmail(String oldEmail) {
    int  k = s.indexOf('_');
    if  (k < 0)
        k = s.indexOf('.');
    int  at= s.indexOf('@');
    return  s.substring(k+1,at) + "." + s.charAt(0) + "@" + s.substring(at+1,s.length()-3) + "net";
}//fixEmail
```

Uppgift 39.

```
public static String middle(String str) {
    if (str.length() % 2 != 0)
        return str.substring(str.length() / 2, str.length() / 2 + 1);
    else
        return str.substring(str.length() / 2 - 1, str.length() / 2 + 1);

}//middle
```