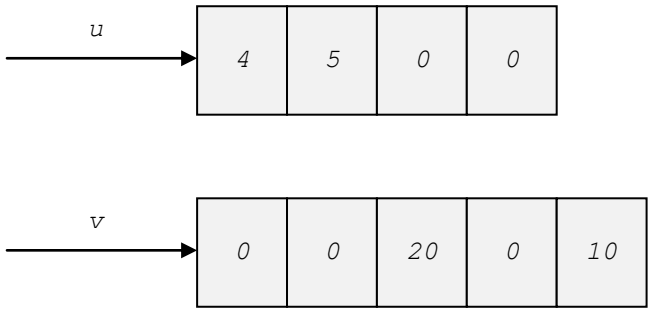
Tentamen – obligatorisk del: lösning

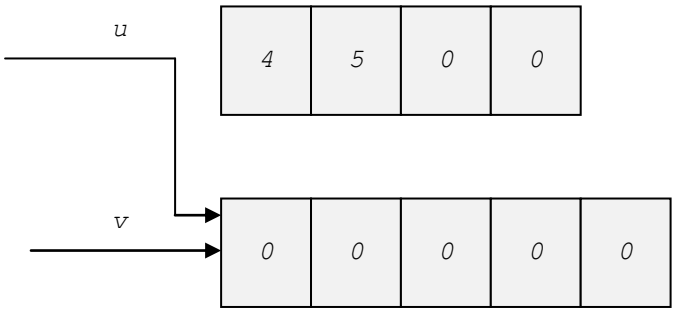
Uppgifter: lösningar

Uppgift 1 (1 poäng + 1 poäng)

a) (1 poäng)



b) (1 poäng)



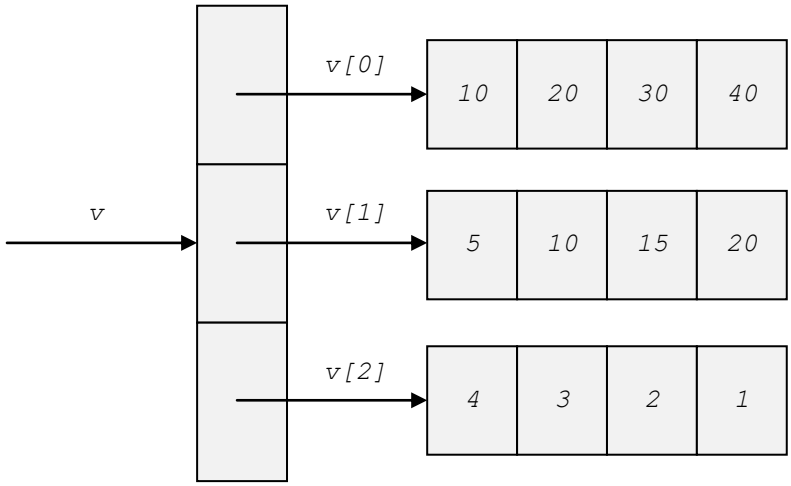
När det givna kodavsnittet har utförts, refererar de båda referenserna till den andra vektorn. Ingen referens refererar till den första vektorn, den kan inte vidare användas, och den blir mål för skräpsamlaren (eng. *garbage collector*).

Uppgift 2 (1 poäng + 2 poäng)

a) (1 poäng)

```
int[][]    v = new int[3][4];
v[0][0] = 10; v[0][1] = 20; v[0][2] = 30; v[0][3] = 40;
v[1][0] = 5; v[1][1] = 10; v[1][2] = 15; v[1][3] = 20;
v[2][0] = 4; v[2][1] = 3; v[2][2] = 2; v[2][3] = 1;
```

b) (2 poäng)



Uppgift 3 (2 poäng + 1 poäng)

a) (2 poäng)

```
// bisektris tar emot längder av två sidor i en triangel och vinkeln
// (i radianer) mellan dessa sidor. Metoden returnerar längden av den
// motsvarande bisektrisen - den som delar den givna vinkeln i två
// lika delar.
public static double bisektris (double b, double c, double alfa)
{
    double    p = 2 * b * c * Math.cos (alfa / 2);
    double    bis = p / (b + c);

    return bis;
}
```

Kortare:

```
public static double bisektris (double b, double c, double alfa)
{
    return (2 * b * c * Math.cos (alfa / 2)) / (b + c);

}
```

b) (1 poäng)

```
double    bis = bisektris (4, 4, Math.PI/2);
```

Uppgift 4 (2 poäng + 2 poäng)

a) (2 poäng)

```
// max tar emot en icke-tom heltalsvektor, och returnerar
// det största heltalet i denna vektor
public static int max (int[] v)
{
    int    m = v[0];
    for (int i = 1; i < v.length; i++)
        if (v[i] > m)
```

```
            m = v[i];

    return m;

}
```

b) (2 poäng)

```
// summa tar emot en icke-tom, tvådimensionell vektor med heltal,
// och returnerar heltalens summa
public static int summa (int[][] v)
{
    int    s = 0;
    for (int i = 0; i < v.length; i++)
        for (int j = 0; j < v[i].length; j++)
            s += v[i][j];

    return s;

}
```

Uppgift 5 (3 poäng)

i	x_i	m	x_m
1	5	1	5
2	8	1	5
3	9	1	5
4	4	4	4
5	7	4	4
6	6	4	4
7	2	7	2
8	10	7	2

Uppgift 6 (2 poäng + 1 poäng)

a) (2 poäng)

```
// sifferSumma tar emot en icke-tom teckensträng, som bara
// innehåller siffror. Metoden returnerar summan av alla
// dessa siffror.
public static int sifferSumma (String s)
{
    int    summa = 0;
    int    len = s.length ();
    for (int i = 0; i < len; i++)
        summa += s.charAt (i) - 48;

    return summa;

}
```

b) (1 poäng)

```
String    s = "764399750102355";
int    summa = sifferSumma (s);
```

Uppgift 7 (3 poäng)

```
Bbbb
AaaaaA
```

Uppgift 8 (2 poäng + 1 poäng + 2 poäng)

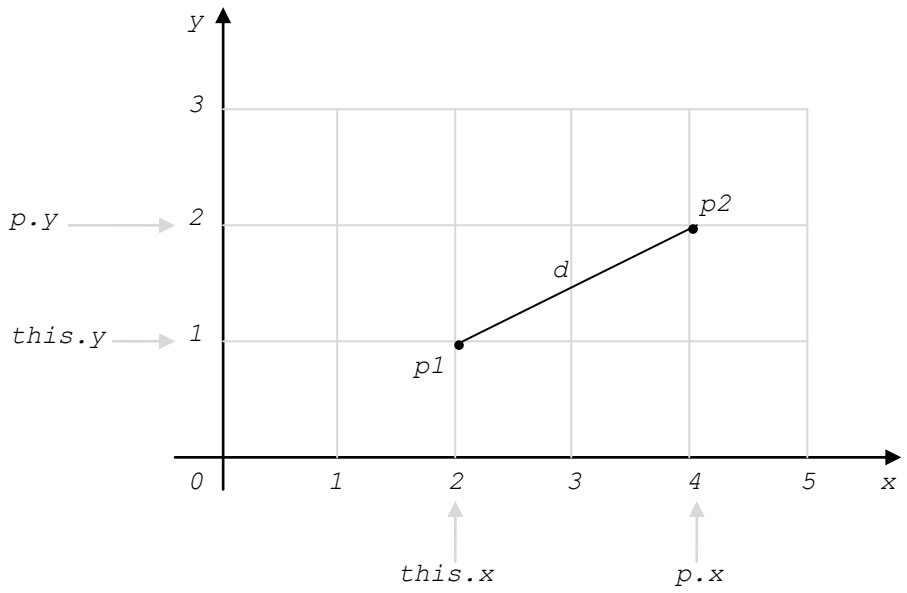
a) (2 poäng)

```
// avstand returnerar avståndet mellan punkten och en given punkt
public double avstand (Punkt p)
{
    double    d = 0;

    d = Math.sqrt ((p.x - this.x) * (p.x - this.x)  +
                   (p.y - this.y) * (p.y - this.y));

    return d;
}
```

b) (1 poäng)

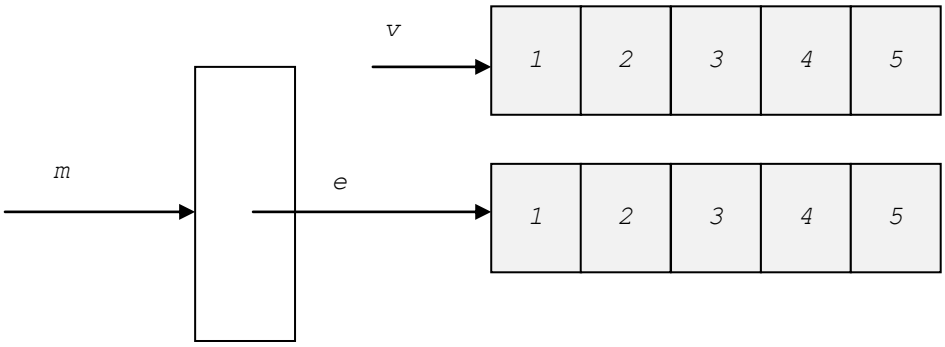
```
Punkt    p1 = new Punkt (2, 1);
Punkt    p2 = new Punkt (4, 2);

double    d = p1.avstand (p2);
```

c) (2 poäng)

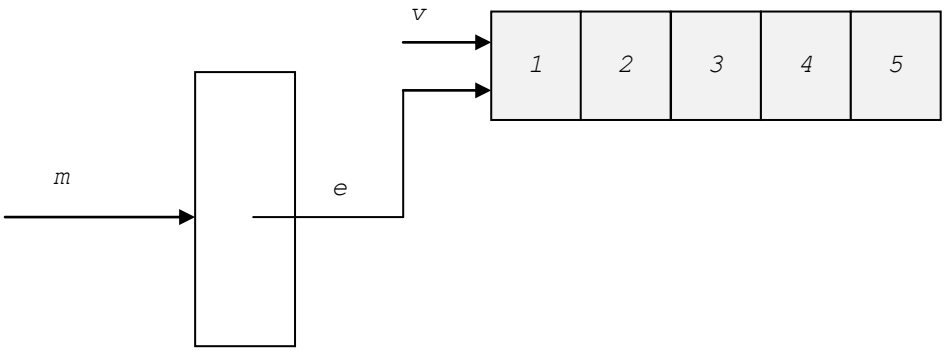


Uppgift 9 (2 poäng + 2 poäng)

a) (2 poäng)



b) (2 poäng)



Uppgift 10 (1 poäng + 2 poäng)

a) (1 poäng)

Vektorn (som refereras av referensen) `a` innehåller referenser som kan referera till objekt av typen `A`. Det betyder att de kan även referera till objekt av olika subtyper till typen `A`. Typen `B` är en subtyp till `A` (eftersom `class B extends A`), och därför kan referenserna i vektorn referera även till objekt av den typen.

b) (2 poäng)

```
4
5
10
11
```

När ett objekt anges som argument i metoden `println`, anropas först dess metod `toString`, och därefter skrivs den returnerade teckensträngen ut. Både klassen `A` och klassen `B` har metoden t`oString`.

Vilken variant av metoden `toString` som ska anropas, beror på typen av det refererade objektet (polymorfism). När referensen `a[i]` refererar till ett objekt av typen `A`, aktiveras den metod som finns i klassen `A`. Då skrivs ut det heltal som inkapslas i objektet. När referensen `a[i]` refererar till ett objekt av typen `B`, aktiveras den metod som finns i klassen `B`. I så fall skrivs ut det inkapslade heltalet utökat med `1`.

Objekten på index `0` och index `2` är av typen `A`: då skrivs det inkapslade heltalet. Objekten på index `1` och index `3` är av typen `B`: då skrivs ut det inkapslade heltalet utökat med `1`.