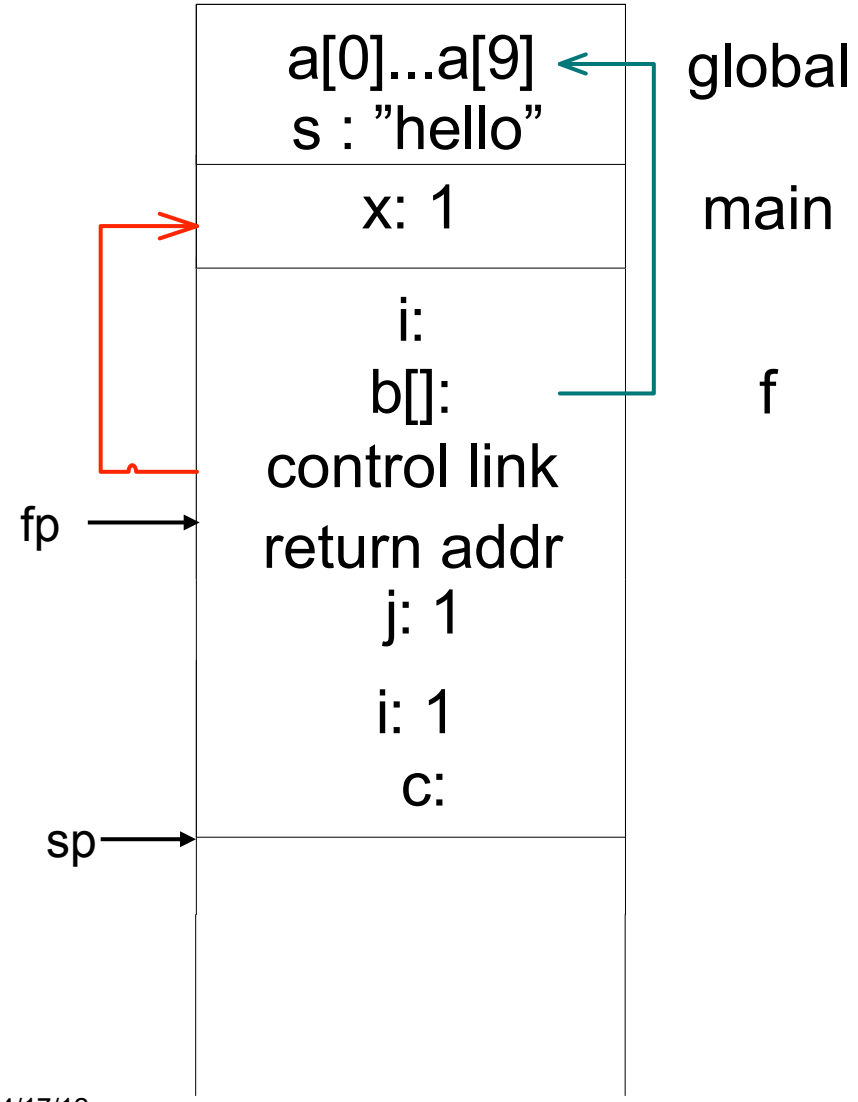
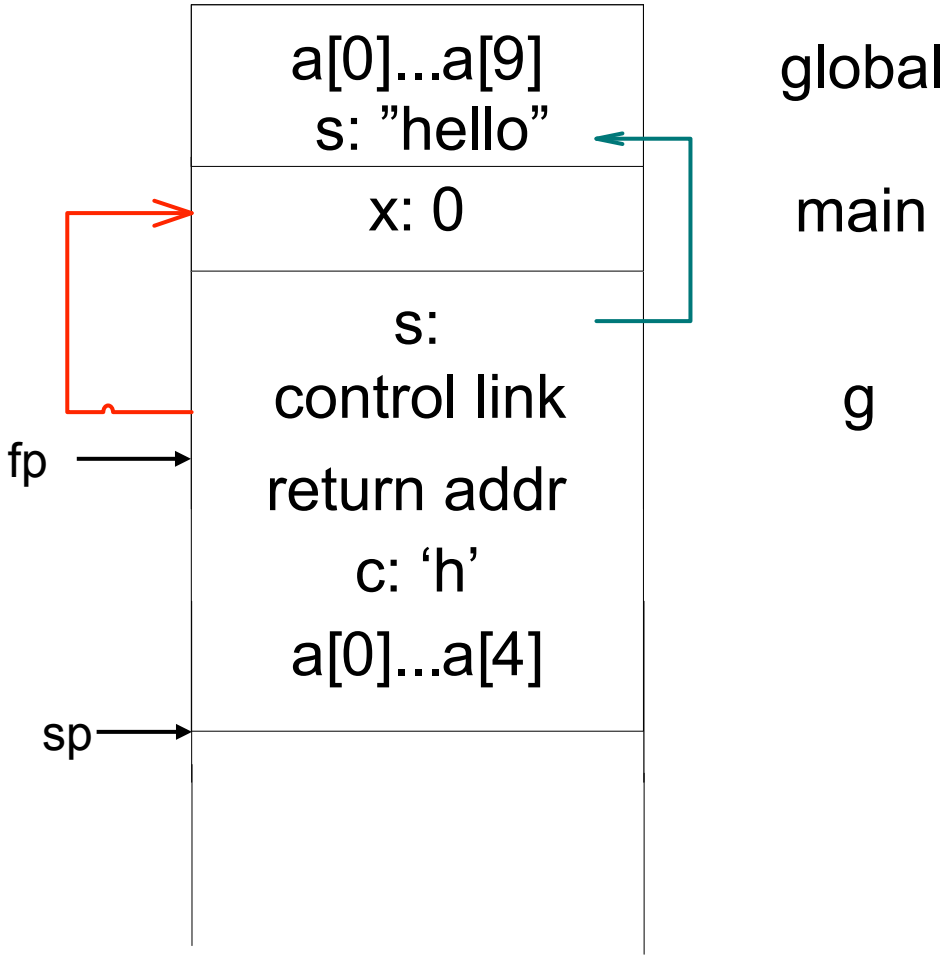
7.2

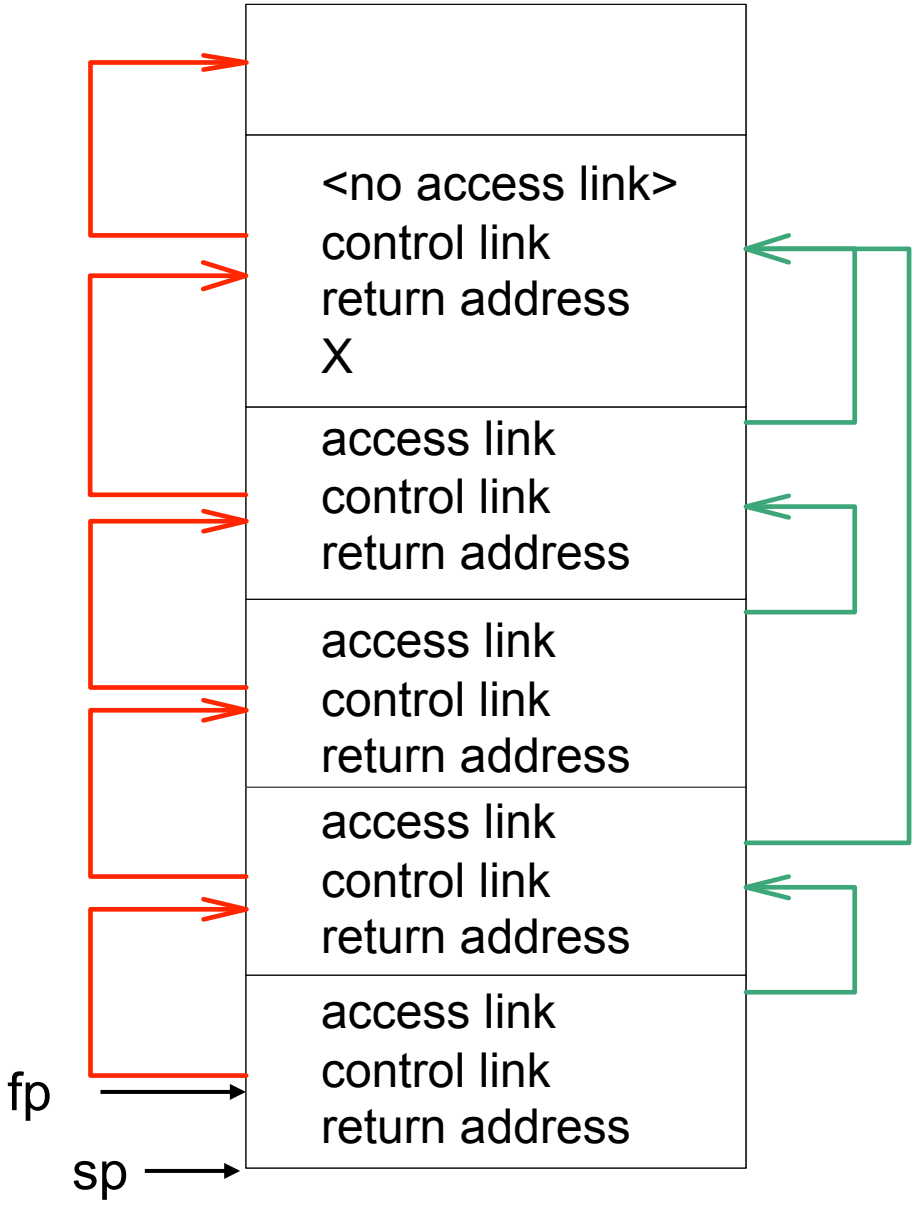
a.


a[0]...a[9]
s : ”hello”
global
x: 1
main
i:
b[]:
f
control link
fp
return addr
j: 1
i: 1
c:
sp

b.


a[0]...a[9]
s: ”hello”
global
x: 0
main
s:
control link
g
fp
return addr
c: ‘h’
a[0]...a[4]
sp

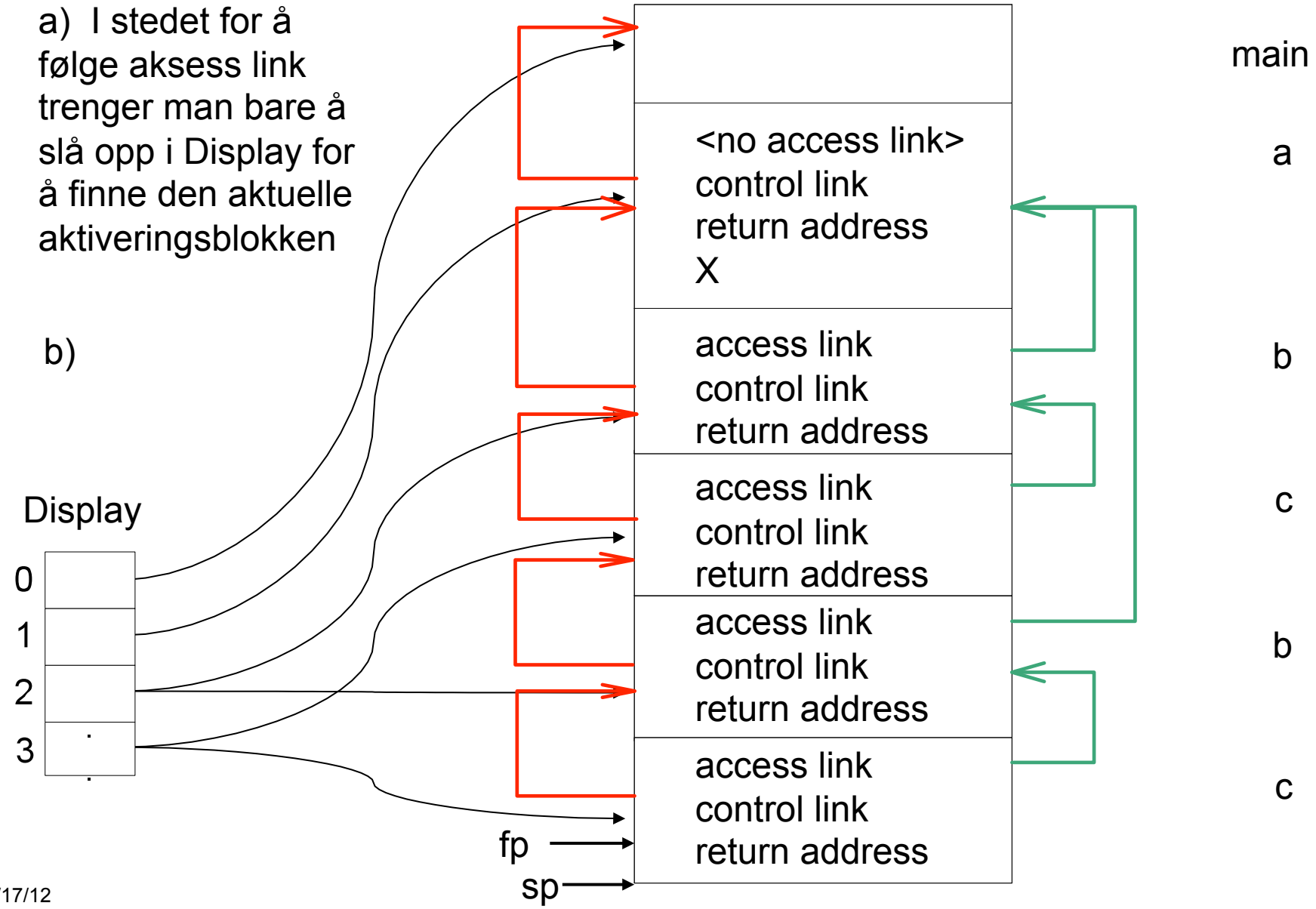
7.4


<no access link>
control link
return address
X
access link
control link
return address
access link
control link
return address
access link
control link
return address
access link
control link
return address
fp
sp
main
a
b
c
b
c

7.10

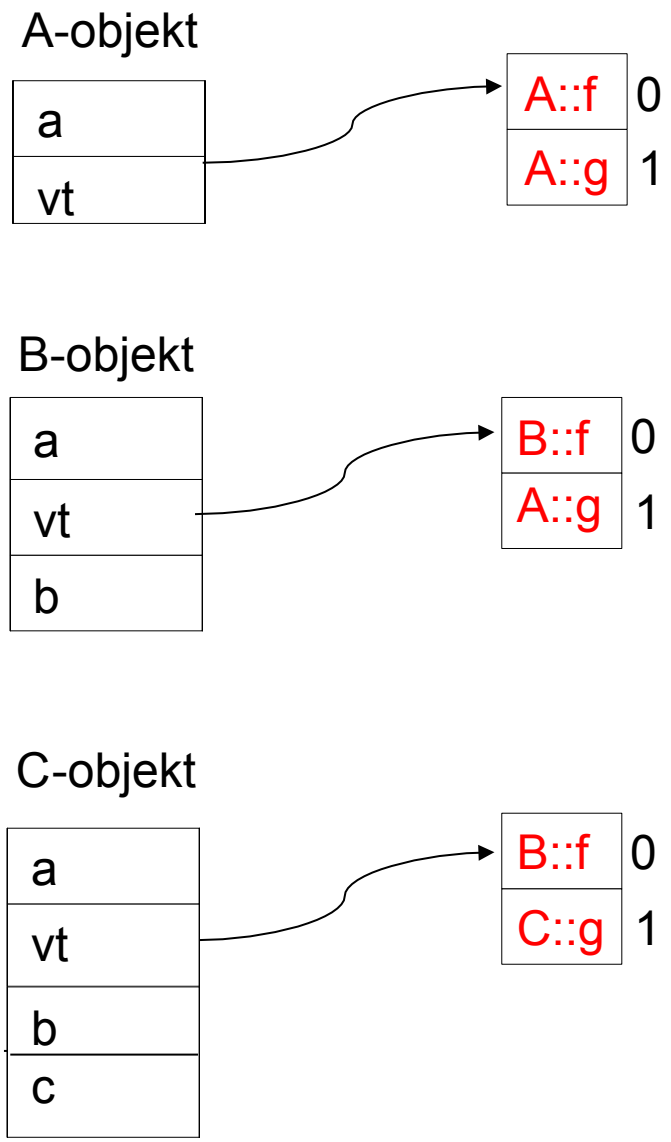
- a) I stedet for å følge aksess link trenger man bare å slå opp i Display for å finne den aktuelle aktiveringsblokken

b)



7.13 Draw the memory layout of objects of the following C++ classes, together with the virtual function tables as described in Section 7.4.2:

```
class A
{ public:
  int a;
  virtual void f();
  virtual void g();
};
class B : public A
{ public:
  int b;
  virtual void f();
  void h();
};
class C : public B
{ public:
  int c;
  virtual void g();
}
```

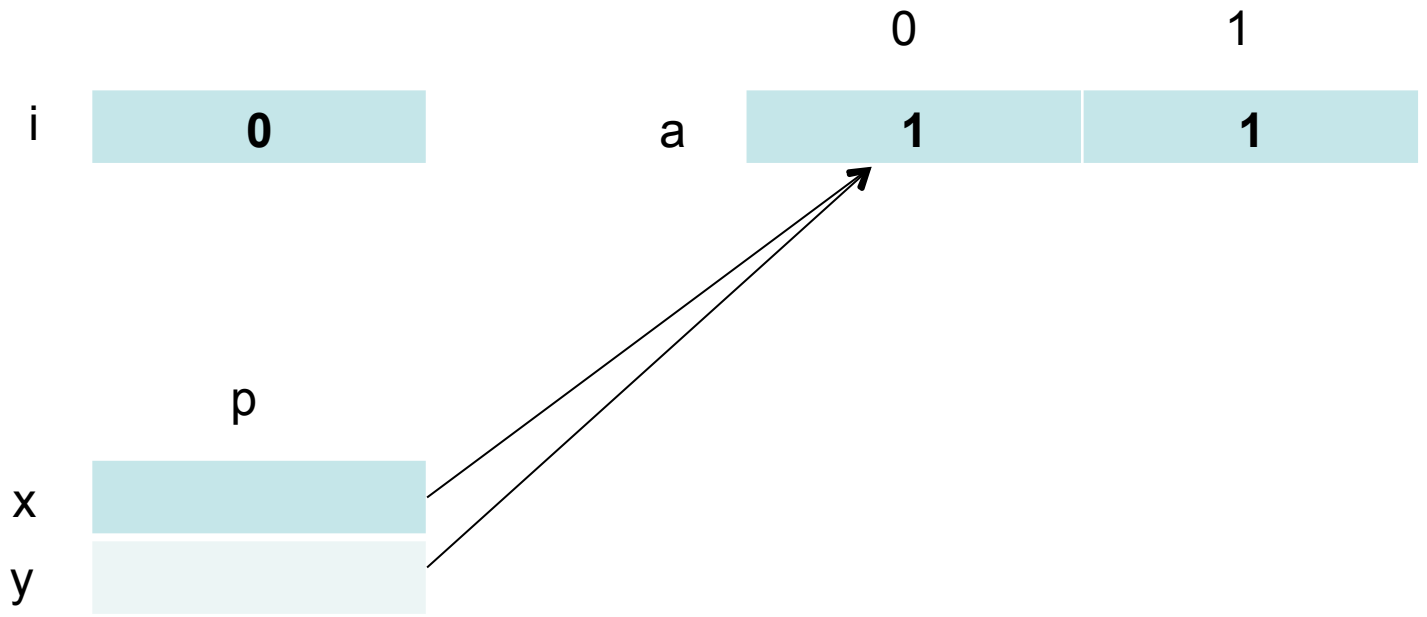


7.15 Give the output of the following program (written in C syntax) using the four parameter passing methods discussed in Section 7.5:

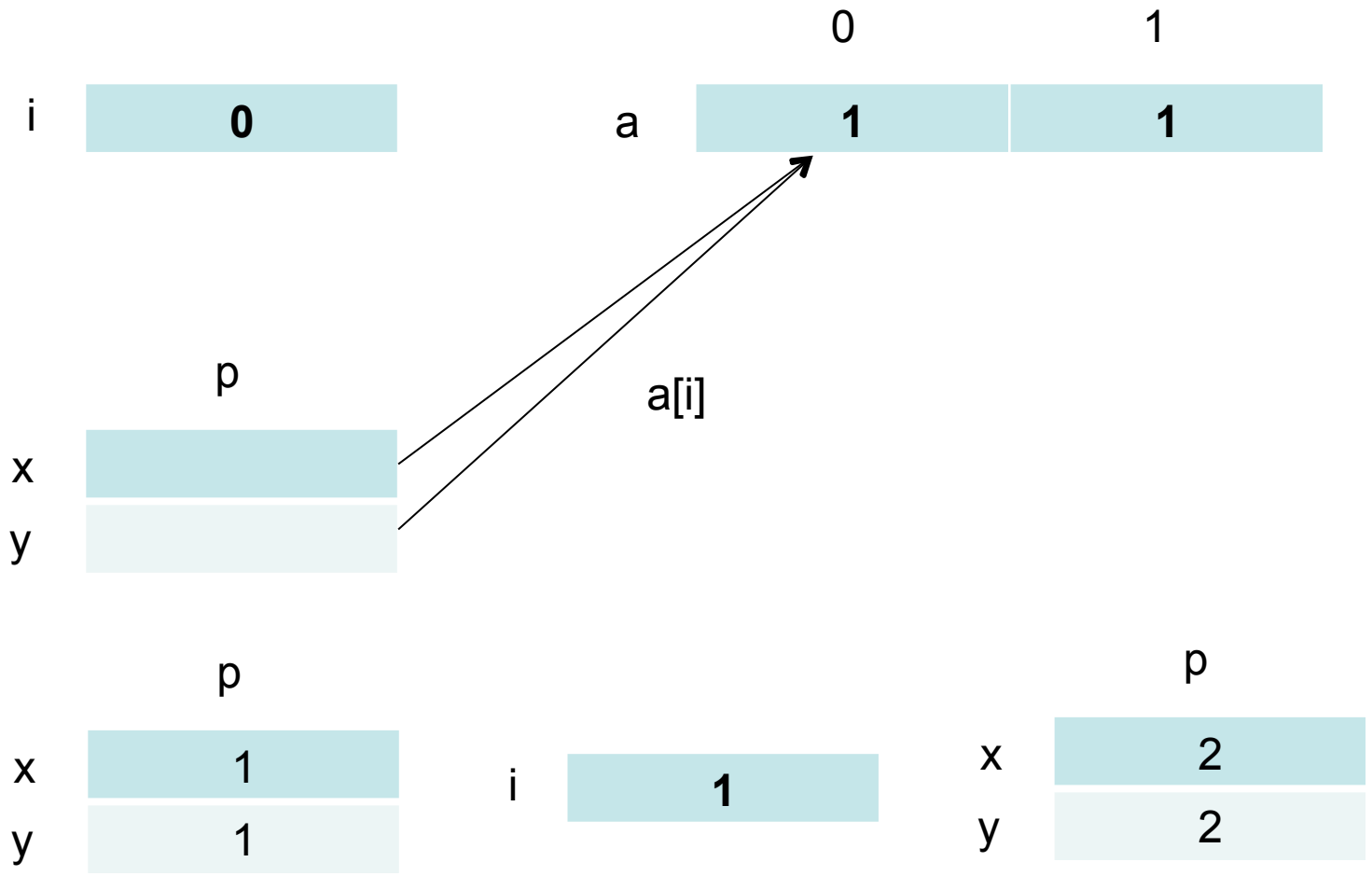
```
#include <stdio.h>
int i=0;

void p(int x, int y)
{ x += 1;
  i += 1;
  y += 1;
}

main()
{ int a[2]={1,1};
  p(a[i],a[i]);
  printf("%d %d\n",a[0],a[1]);
  return 0;
}
```

by value	by reference
1 1	3 1
by value-result	**by name**
2 1 1 2	2 2

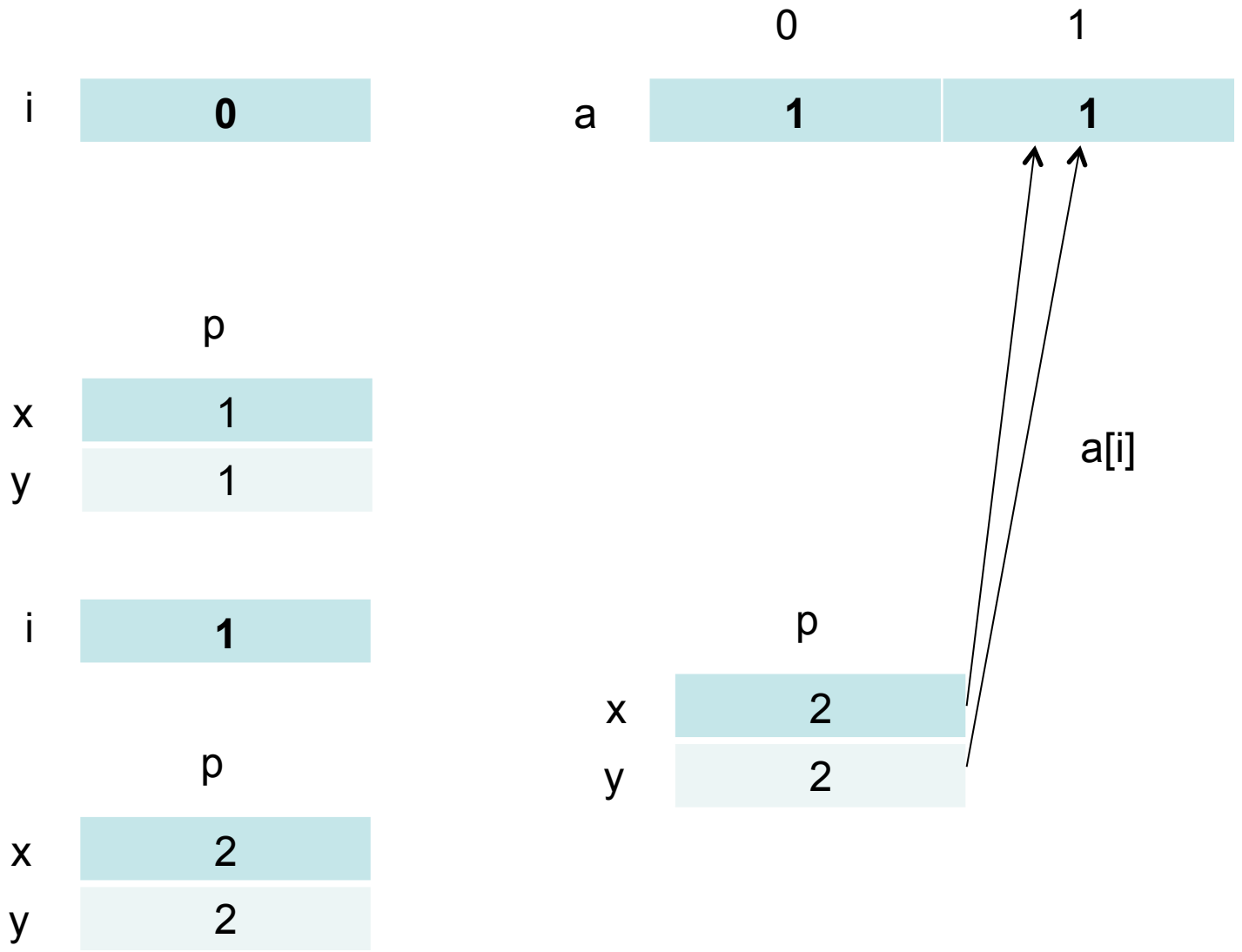
7.15 by reference



7.15 by value-result – address at call



7.15 by value-result – address at exit



7.15 by name

a(i) = a(i) + 1 == a(0) = a(0) + 1 = 1 + 1 = 2

i = i +1 == i = 0 +1 = 1

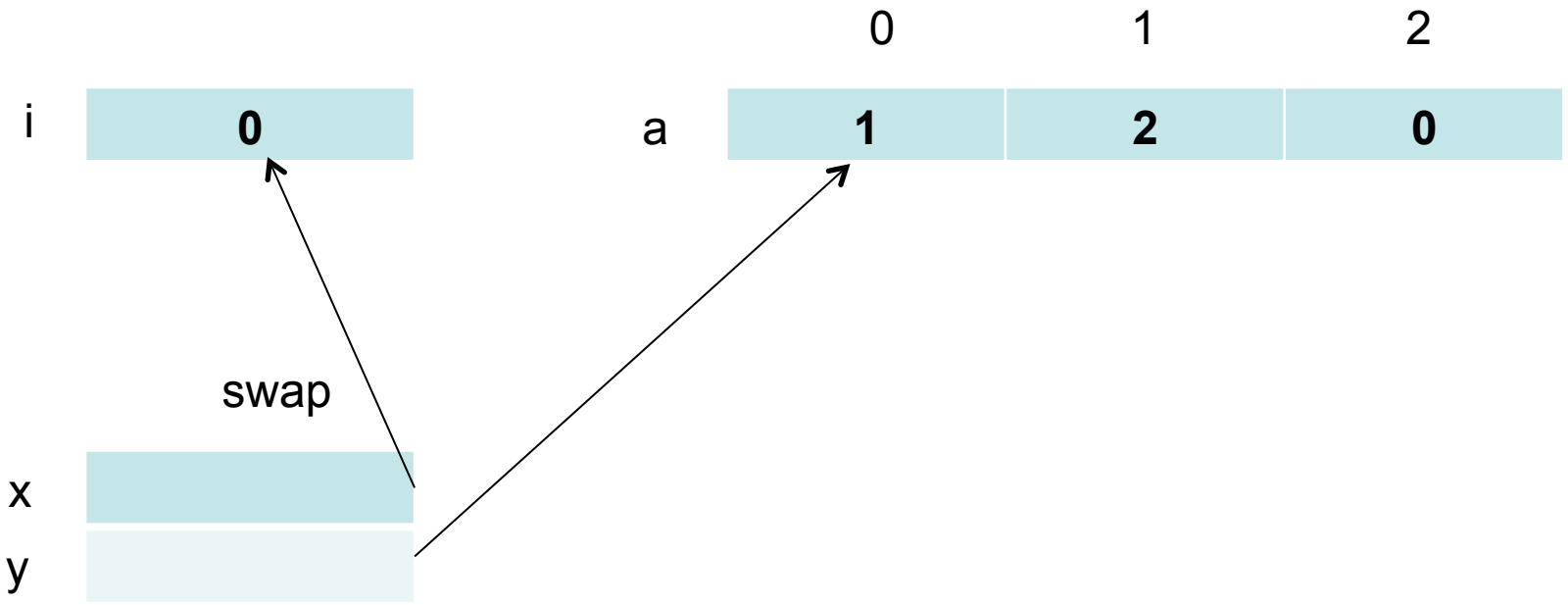
a(i) = a(i) + 1 == a(1) = a(1) + 1 = 1 + 1 = 2

7.16 Give the output of the following program (in C syntax) using the four parameter passing methods of Section 7.5:

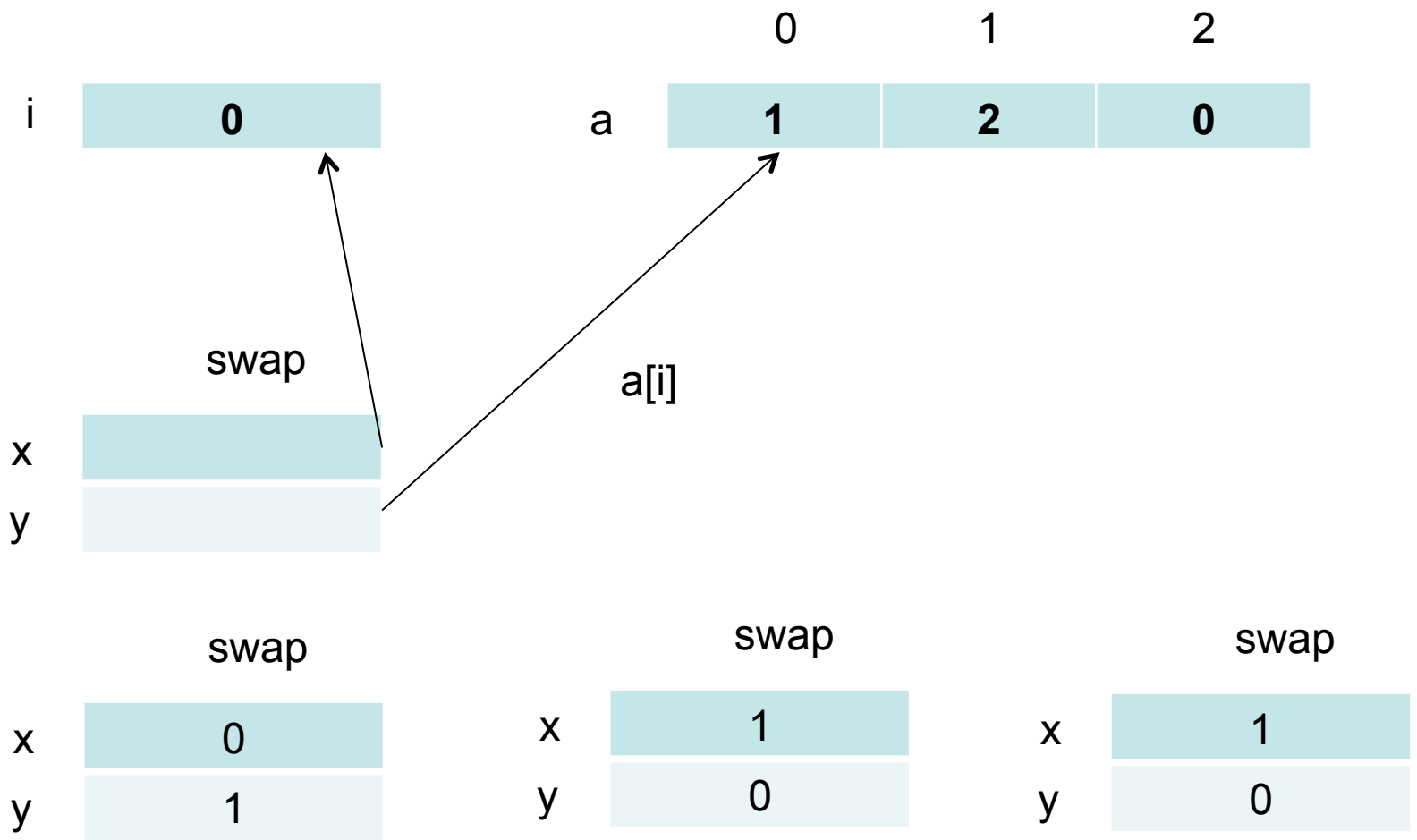
```
#include <stdio.h>
int i=0;

void swap(int x, int y)
{ x = x + y;
  y = x - y;
  x = x - y;
}

main()
{ int a[3] = {1,2,0};
   swap(i,a[i]);
   printf("%d %d %d %d\n",i,a[0],a[1],a[2]);
   return 0;
}
```

by value	by reference
0 1 2 0	1 0 2 0
by value-result	by name
1 0 2 0 1 1 0 0	2 1 -1 0

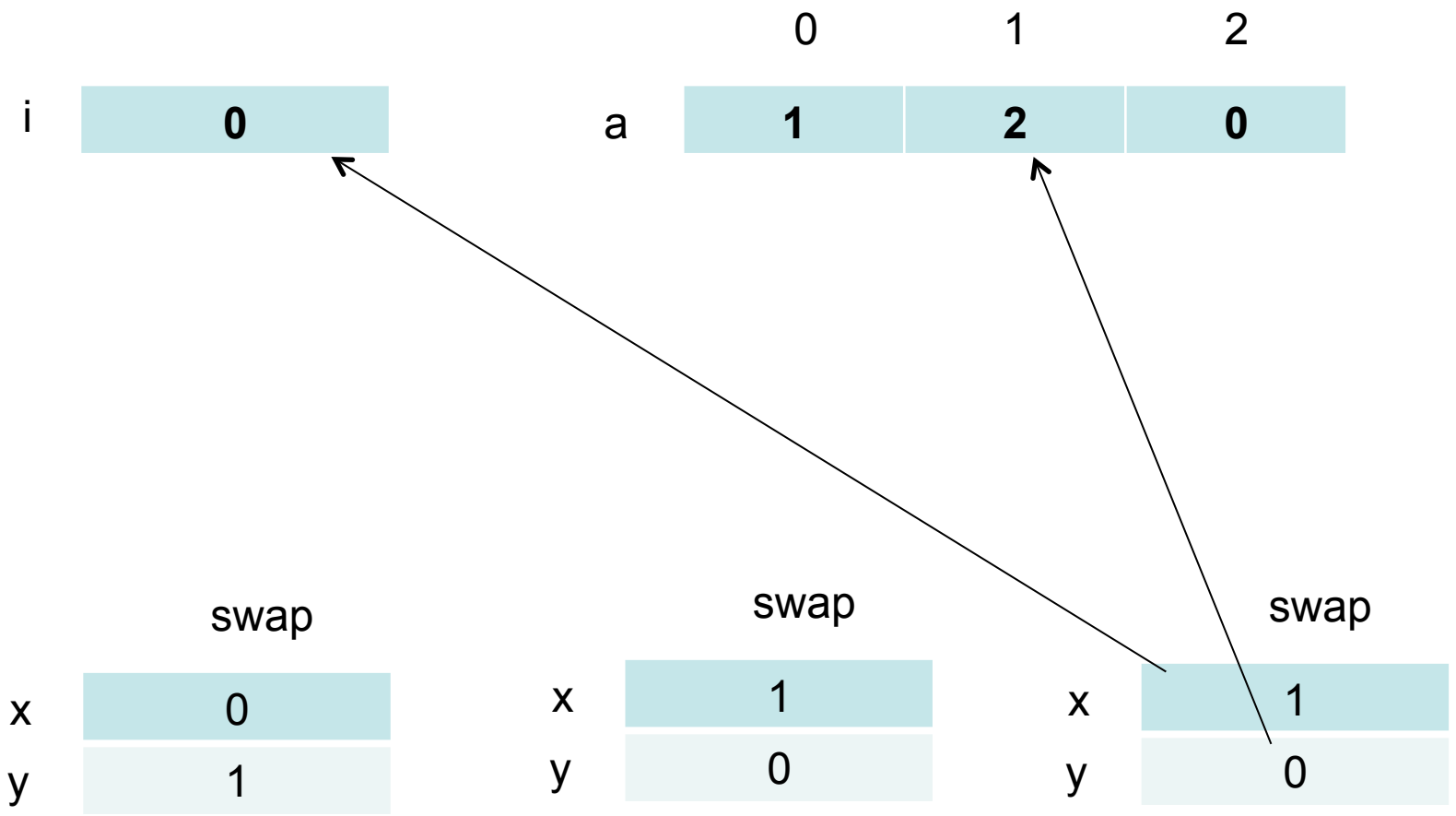
7.16 by reference



7.16 by value-result – address at call



7.16 by value-result – address at exit

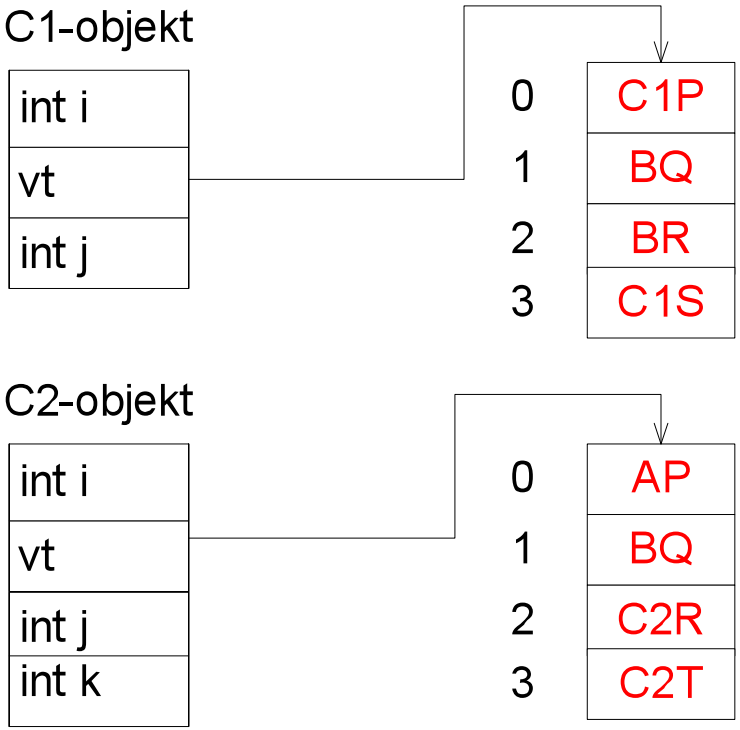


7.16 by name

i = i + a(i)	==	i= 0 + 1 = 1		
a(i) = i - a(i)	==	a(1) = 1 – a(1)	==	a(1) = 1 – 2 = -1
i = i - a(i)	==	i = 1 – a(1)	==	i = 1 – (-1) = 2

Eksamen 2005

a)



Eksamen 2005

b)

Vi finner nå på å innføre i språket muligheten for å spesifisere en metode til å være **final**. Det skal bety at den ikke lenger er virtuell, dvs at den ikke kan redefineres i subklasser.

Anta at vi i klassen B spesifiserer metoden Q til å være **final**.

Må vi da endre på virtuell-tabellen for B-objekter?

Begrund svaret.

NEI, virtuelle metoder i B-objekter kan fremdeles kaldes via A-typede pekere.

Eksamen 2005

c)

```
instanceOf = false;
cd= <refExpr>.vt.cl;
while not(cd= klassedeskriptor for klassen Object) do
  { if cd= klassedeskriptor for klassen <class>
    then instanceOf= true;
    cd= cd.super
  }
```

Eksamen 2005

d)

Generell test:

- Sjekk at <class>.subklassenivå ligger innenfor grensene på den aktuelle supers-tabellen
- <refExpr>.vt.cl.supers[<class>.subklassenivå] = <class>

Konkrete tester:

```
rc11.vt.cl.supers(3) = C1   dvs C1 = C1    dvs true
rc11.vt.cl.supers(3) = C2   dvs C1 = C2    dvs false
```

Eksamen 2005

d) Supers for C11 og C21

	Object
0	
1	A
2	B
3	C1
4	C11

	Object
0	
1	A
2	B
3	C2
4	C21