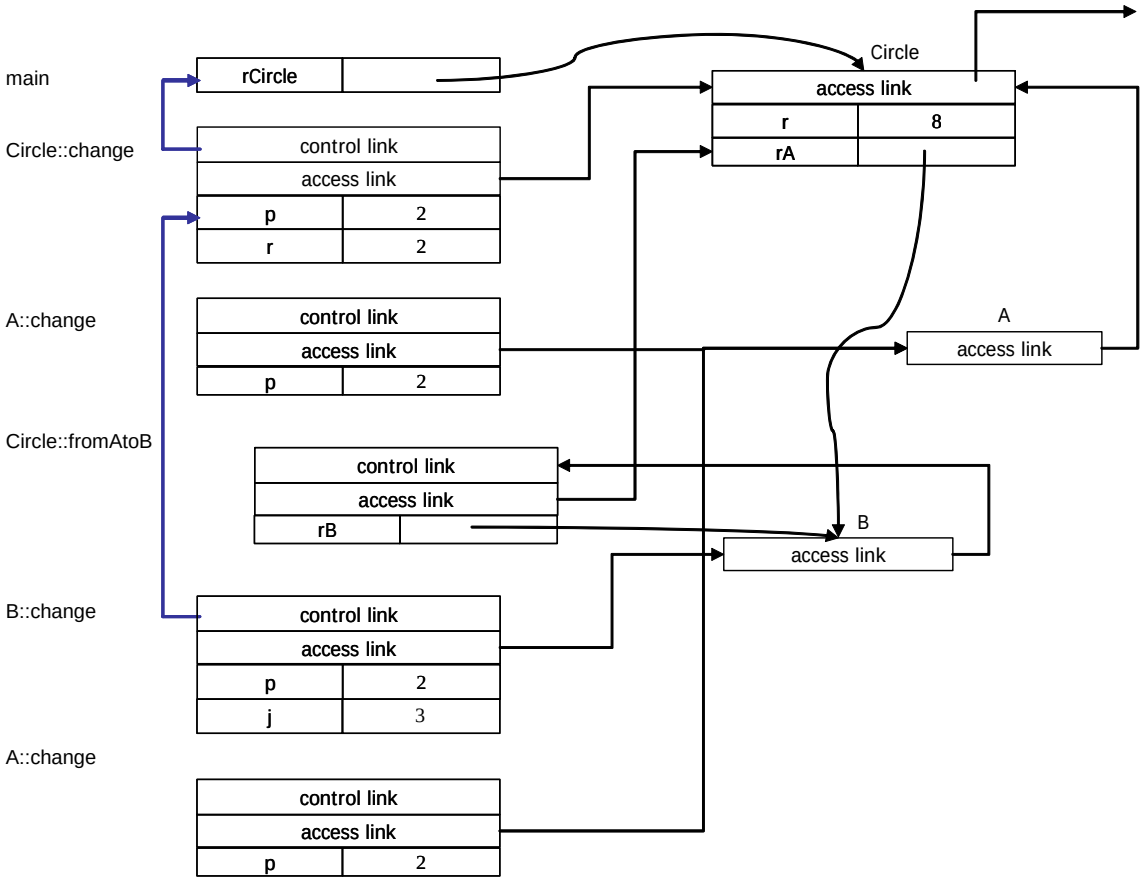
Oppgave 1

1a



1b

a)

Ja

Typen til resultatet av `getCenter` er garantert å være enten `Point` eller `ColoredPoint`, og disse kan begge assignes til `somePoint`, som har typen `Point`.

b)

Point

Referansen `somePoint` kan peke ut et `Point` object, derfor må `setCenter` i `ColoredCircle` kunne akseptere parametre av typen `Point`.

Oppgave 2

2a

a)

```
fun max(i, j) = if i >= j then i else j ;
```

b)

```
(* Rett frem  *)
fun maxlist([])  = 0
  | maxlist(x::xs) = max(x,maxlist(xs)) ;

(* Samme med case syntaks *)
fun maxlist(il)  =
    case il of nil => 0
         | x::xs => max(x,maxlist(xs));

(* med foldl/foldr *)
fun maxlist(xs) = foldl (max) 0 xs ;
fun maxlist(xs) = foldr (max) 0 xs ;

(* en variant med hjelpefunksjon  *)
fun maxl([],n) = n
  | maxl(x::xs,n) = if x > n then maxl(xs,x) else maxl(xs,n) ;
fun maxlist(xs) = maxl(xs,0) ;

(* og enda en  *)
fun maxlist([])  = 0
  | maxlist([x]) = x
  | maxlist(x::y::xs) = if x > y then maxlist(x::xs) else maxlist(y::xs);
```

c)

```
fun sumlist([]) = 0
  | sumlist(x::xs) = x+sumlist(xs);

(* med case-uttrykk  *)
fun sumlist(il) =
    case il of nil => 0
         | x::xs => x+sumlist(xs);


(* med fold  *)
fun sumlist(xs) = foldl (op+) 0 xs ;
fun sumlist(xs) = foldr (op+) 0 xs ;


(* med hjelpefunksjon  *)
fun suml([],n) = n
  | suml(x::xs,n) = suml(xs,(x+n));
fun sumlist(xs) = suml(xs,0) ;
```

2b

```
fun mapFirstTwo f [] = []
  | mapFirstTwo f [x] = []
  | mapFirstTwo f (x::y::xs) = f(x,y) :: (mapFirstTwo f (y::xs)) ;
```

2c

a)

```
val fa = Term("f",[Term("a",[])]);
val fxyz = Term("f",[Term("x",[]),Term("y",[]),Term("z",[])]);
val fgh =
Term("f",[Term("g",[Term("a",[]),Term("b",[])]),Term("h",[Term("c",[]),Term("
d",[])])]);
```

b)

```
fun height(Term(n,[])) = 1
  | height(Term(n,ts)) = 1 + maxlist (map height ts);

(* Mange varianter er mulig også her  *)
```

Oppgave 3

3a

```
enkel spørring:

?- res(X,inf3110,Y)

eller med regel og spørring:

taken(X,Y) :- res(X,Y,_Z) .
?- taken(X,inf3110)
```

3b

```
spørring uten regel;
?- res(X,inf2200,Z), Z \== f .

eller med regel:

passed(X,Y) :- res(X,Y,Z), Z \== f .

spørring:
?- passed(X,inf2200) .
```

3c

```
- ikke samme kurs og person to ganger
- resultat må være  a,d,c,d,e,f

multi(X,Y) :-  res(X,Y,Z), res(X,Y,Z1) , Z\==Z1 .
gradeOK :- res(_X,_Y,Z) , legal(Z) .
legal(Z) :- Z==a ; Z==b ; Z==c ;Z==d ; Z==e ; Z==f .
faktaOK :- \+ multi(_X,_Y) , gradeOK .
```

3d

```
hjelperelasjon:

moreThanOneApplicant(X,Y) :- applied(X,Y), applied(X1,Y) , X1 \== X  .

X er en gyldig kandidat til jobben som gruppelærer:

candidate(X,Y) :- applied(X,Y) , passed(X,Y) , \+ ta(X,Y,_Z) .
candidate(X,Y) :- applied(X,Y) , passed(X,Y) , \+ moreThanOneApplicant(X,Y) .
```