Esercizio 1 Soluzione

a) in x=5; y= 10
   out:5
b) Dobbiamo usare AcodeE:
   in x=5; y= 10
   Errore in "y =z+y;" y è legato ad un valore denotabile che non è un l-valore
c) Errore in "foo(x+x,x);" non è un l-valore.
d) in x=5; y= 10
   out:15
e) in x=15; y= 10
   out:15

Esercizio 2 Soluzione

a)
```
let rec map = fun f l -> match l with
                |[] -> []
                |x::lR -> (f x) :: (map f lR)
                ;;
   map (fun x -> x * x) l;;
```
b)
```
let rec mapT = fun f l r -> match l with
                |[] -> r
                |x::lR -> mapT f lR (r @ [(f x)])
                ;;
   List.flatten (mapT (fux x -> if (x > 0) then [x] else []) l []);;
```
c)
```
let mapI = fun f l -> List.fold_right (fun x u -> (f x)::u) l [];;
```

Esercizio 3 Soluzione

a)
```
public class Fun2 <A> extends Fun<A,A>{
    public boolean fix(int k){
     LinkedList<A> d = dom();
     int count = 0;
     for(A x : d){
           if(x.equals(apply(x))) count++;
     };
     return(!(count > k));}}
```
b)
```
public class Fun3k<A> extends Fun<A,A>{
    int maximum;
    int current;
  public Fun3k(int k){
    super();
    maximum = k;
    current = 0;
    }
  public void add(A x, A y){
    if (x.equals(y)){
     if (isIn(x)) {
           if(y.equals(apply(x))) return;
           if(current<maximum){super.add(x,y); current++;return;};
           return;};
     if(current<maximum){super.add(x,y); current++; return;};
     return;};
    super.add(x,y);}

  private boolean isIn(A x){
    return dom().contains(x);}}
```

c)
```
public class Fun4<A extends Comparable<A>> extends Fun3k<A>{
  public Fun4(int k){
   super(k);}
  private boolean isMono(){
   LinkedList<A> d = dom();
   for(A i: d) {
     for (A j: d){
      if((i.compareTo(j)<=0) && (apply(i).compareTo(apply(j))>0))
         return false;}}
   return true;}}
```