Es. 1

S → A D
A → 1 | 1A | B
B → 2 | 2B | C
C → 3 | 3C
D → 3 | 3C | E
E → 2 | 2E | F
F → 1 | 1F

Es. 2

```
let rec valdepth bt = match bt with
        Void -> (0, 0)
  | Node (x, Void, Void) ->  (x, 1)
  | Node (x, lbt, rbt) when lbt <> Void or rbt <> Void
        -> let (v1, p1) = valdepth lbt
           and (v2, p2) = valdepth rbt
           in
           if v1 > v2 then (v1, p1+1)
                      else (v2, p2+1);;
```

Es. 3

```
int check ( int a[], int dim)
{ int i = 0;
  int verificato = 1;
  while (i < dim && verificato)
  {   int esisteminore = 0,
      int tuttiminoriouguali = 1;
      int j = i+1;
      while (j<dim && tuttiminoriouguali)
         if (a[i] < a[j]) tuttiminoriouguali = 0;
         else { if (a[i] > a[j] && ! esisteminore)
                   esisteminore = 1;
                j = j+1;
              }
      if (tuttiminoriouguali && esisteminore)
          i = i + 1;
      else verificato = 0;
  }
  return verificato;
}
```

Es. 4

```
void azzera (int a[], int dima, int b[], int dimb)
{ int i;
  for (i=0; i<dima; i=i+1)
  { int j=0; int trovato=0;
    while (j<dimb && !trovato)
      if (a[i]=b[j]) trovato=1;
      else j=j+1;
    if (!trovato) a[i]=0;
  }
}
```

Es. 5

```
let DelOutside l (x,y) =
    let f (w,z) ys = if w>=0 & z>=0 &
                        w<=x & z<=y
                     then (w,z)::ys
                     else ys
    in foldr f [] l;;
```

oppure

```
let DelOutside l (x,y) =
    let p (w,z) = (w>=0 & z>=0 & w<=x
                   & z<=y)
    in filter p l;;
```

Es 6

```
void scambia (listaDiElementi *l)
{ if (*l != NULL)
    if (*l -> next != NULL)
      if (*l -> next -> next == NULL)
      {
        *l -> next -> next = *l;
        *l = *l -> next;
        *l -> next -> next = NULL;
      }
    else
    {
        listaDiElementi prec = *l;
        listaDiElementi corr = *l -> next;
        while (corr -> next -> next != NULL)
          {prec = corr; corr = corr -> next;}

        prec -> next = corr -> next;
        corr -> next -> next = corr;
        corr -> next = NULL;
    }
}
```