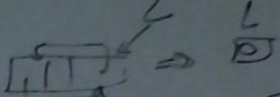
Exo1:

1) struct Element { //selon Applic; };

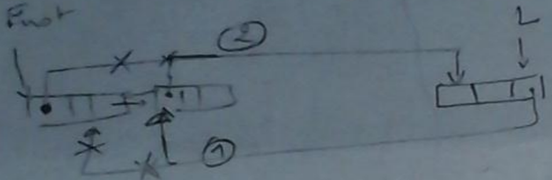
```
struct cel { Element val;
             cel* Pcd;
             cel* suiv; };
typedef cel* Bliste;
```

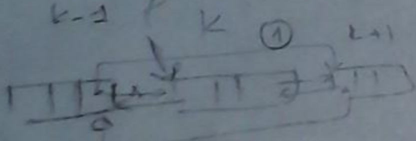
(1pt)

2)




```
Bliste supprimer( Bliste L, int k) {
  int n = longueur (L);
  if (k>=1 and k<= n) {
     if (k==1) if(L->Pcd==L) { delete L; L = null; }   (2pts)
               else { Bliste First = L->suiv;
                      L->suiv = First->suiv;   (1)
                      L->suiv->Pcd = L;   (2)
                      delete First; }          (2pts)
     else { Bliste P=L->suiv;  // P= [illegible]
            for (int i=1; i<k; i++) P=P->suiv;
            P->Pcd->suiv = P->suiv;   (1)
            P->suiv->Pcd = P->Pcd;    (2)
            if (L==P)  L = L->Pcd;    (3)
            delete P; } ;             (2pts)
  return L; }
```

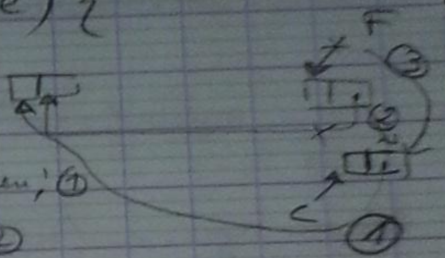


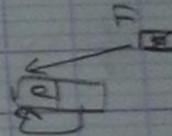


Exo2 :

1)
```
struct Element { ___ };
struct cel { element val;
             cel* lien; };
typedef cel* File;
```
(1pt)

2)
```
File Ajouter ( File F, Element e) {
    cel* c = new cel;
    c -> val = e;
  if (F != null) { c -> lien = F -> lien; ①
                   F -> lien = c;  ②
                   F = c; ③
                 }
    else { c -> lien = c;  F = c; }
return F; }
```
(2pts)





3) Op. fond = Affectation de pointeurs ;

Calcul : $Min_A(n) = 2+1 = 3$

$Max_A(n) = 1+3 = 4$

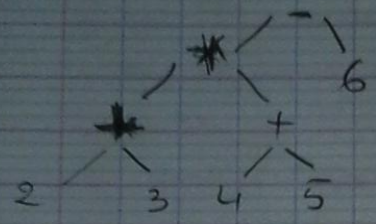
(3/5)

2+3*4+5-6

Exo2 1) $2n+1 = 9 \Rightarrow n = \frac{9-1}{2} = 4$

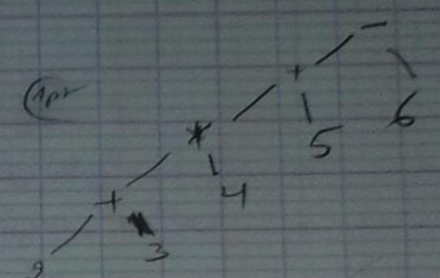
$bc_4 = \frac{1}{4+1} \cdot C_8^4 = \frac{8 \times 7 \times 6 \times 5 \times 4!}{5 \times 4! \times 4 \times 3 \times 2} = \frac{4 \times 2 \times 7 \times 3 \times 2}{4 \times 3 \times 2} = 14$

2)

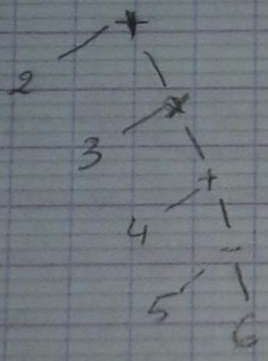


Infixe:

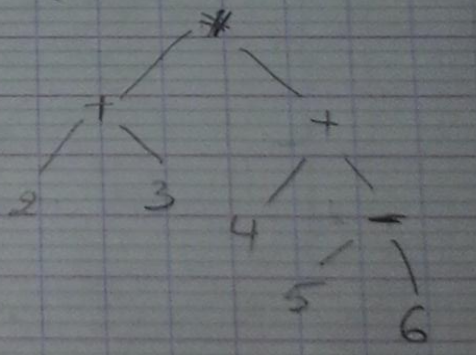
-*+23+456



-+*+23456



+2*3+4-56



*+23+4-56