Algorithmique

Exercice 2.2 :

1) $x = 5 \wedge y = 2 \xrightarrow{y \leftarrow y+1}$ ☹

$x = 5 \wedge y = 2 \xrightarrow{y \leftarrow y+2} x = 5 \wedge y = 3$

2) $x = 5 \wedge y = 2 \xrightarrow{y \leftarrow y+x} x = 5 \wedge y = 7$

3) $x = y + 1 \xleftarrow{y \leftarrow y+1} y = x$

$y = x - 1$

4) $x = y \xrightarrow{y \leftarrow y+1} x = y + 1$

5) $x = y \xrightarrow{y \leftarrow y-1} x = y - 1$

6) $x \neq 0 \wedge y$ est un multiple de $x \xrightarrow{y \leftarrow y-x}$

$x \neq 0 \wedge y = 3 \% x \xrightarrow{y \leftarrow y-x} y \bmod x.$

[ ] partie atteignable $x = y$

[ ] partie atteignable $\Downarrow x \leftarrow x+1$

[+] partie critique.

$x = y + 1$

$k \geq 0 \xrightarrow{k \leftarrow k+1}$ si $x = 0$ $k = 1$

si $k > 0$ $k > 1$

$k \geq 0 \wedge \varphi = k!$

$\varphi = \varphi \times k$

$k \geq 0 \wedge \varphi = (k+1)!$

$k \leq n-1$

$k = k+1$

$k \leq n$

$k \geq 1 \wedge \varphi = (k-1)!$

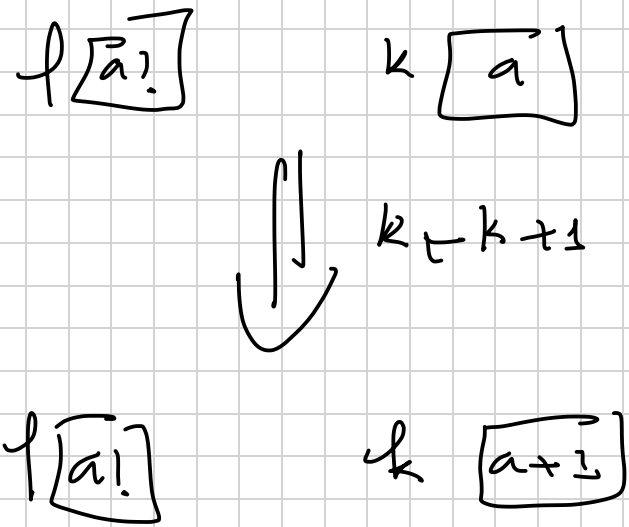
$p = \varphi \times k$

$k \geq 1 \wedge \varphi = (k-1)! \times k = k!$

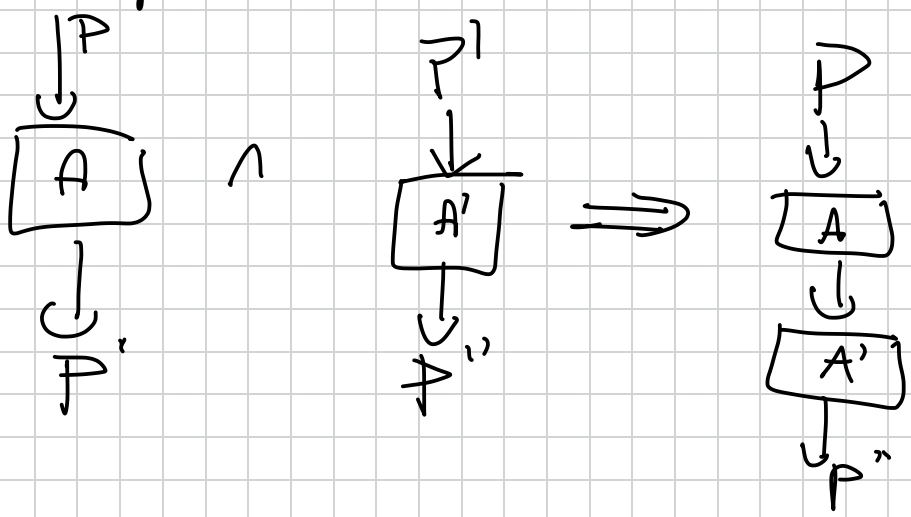
$\varphi = k!$

$k = k+1$

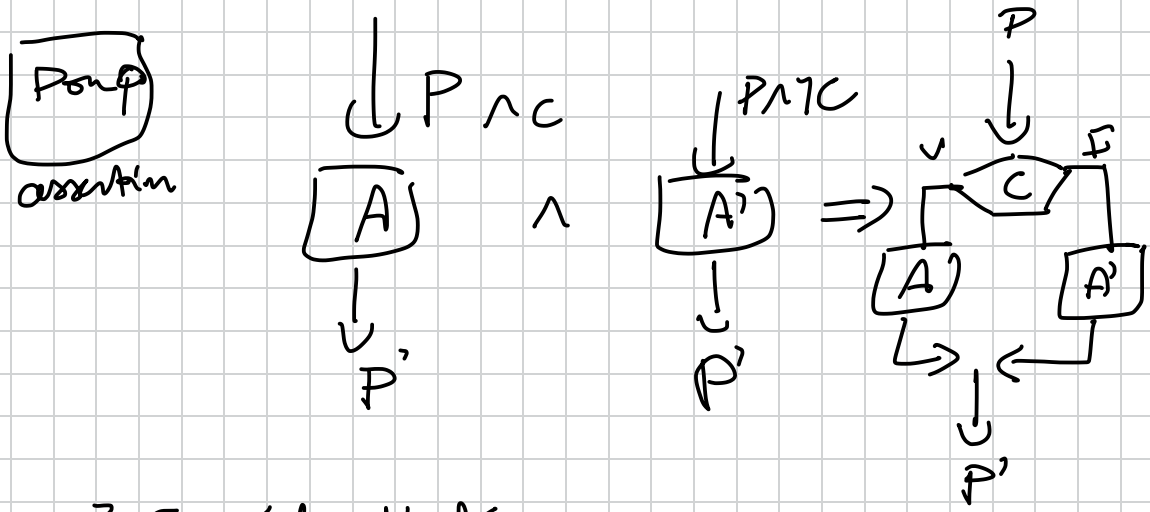
$\varphi = (n-1)!$



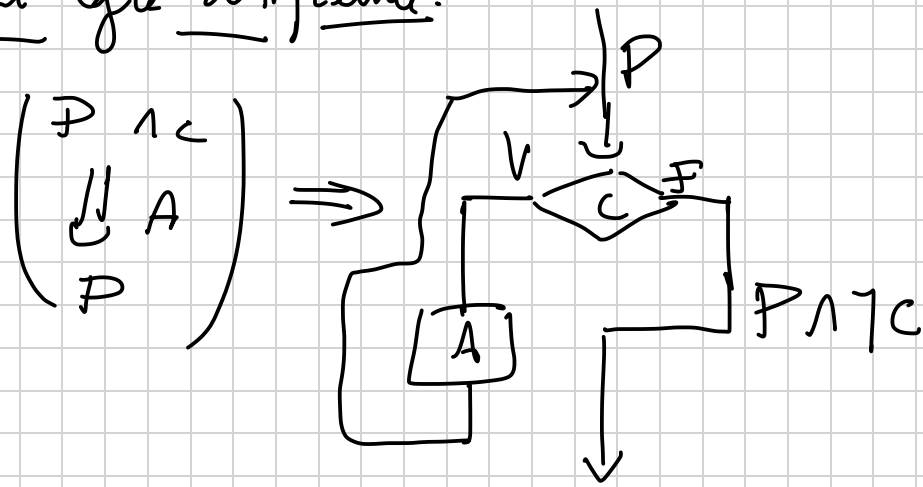
Logique Hoare : (Aide faire programme corrects)
une règle d'inférence :



2ème règle d'inférence :



3ème règle d'inférence :



Pb : Ecrire un algo qui calcul n! / n : donné

F ← 1 k ← 0 Si k ← 0 ; F ← 1 Variable : f

Tant que k ≠ n

k ← k + 1

f ← f × k

retourne F;

k [?] f [?]

φ : $0 \leq k \leq n \wedge f = k!$

formuler un algorithme qui présente la valeur de n entier positif entre la valeur de 1 an

en croissant

k = 1 ;

| tant que k <= n ;
    écrire (k);
k ← k + 1 |

Exercices : 1.1) Eg 1.2) Eg (3,8Mo)

Exemple échange de valeurs.

Décrire une méthode d'échange de contenu de 2 variables x et y de même type.

Analyse :

la solution « temp ← x, x ← y, y ← temp »

il faut prouver la correction.

Spécification :

l'assertion d'entrée de l'algo est : $x = a \wedge$

l'assertion de sortie de l'algo est : $y = b \wedge x = b \wedge y = a$

$$x = a \wedge y = b \xRightarrow{temp \leftarrow x} temp = a \wedge y = b$$

$$temp = a \wedge y = b \xRightarrow{x \leftarrow y} temp = a \wedge x = b$$

$$temp = a \wedge x = b \xRightarrow{y \leftarrow temp} x = b \wedge y = a$$

Preuve de la terminaison :

Etant donné que la séquence est composée d'affectation

réduites à des expressions de contenus, elle se termine.

Exemple: la factorielle.

Problème: étant donné un naturel n, décrire une méthode de calcul de n!.

AE de l'algorithme: P: aucune

AS de l'algorithme: k: f = n!

$$Q: 0 \leq k \leq n \wedge f = k!$$

C: $k < n$

problème: Ranger dans l'ordre croissant une ribambelle d'objets.

Analyse:

la solution: tant que 2 objets ont des places différentes ont fait que celui devant et regarde le fait échanger de place.

```
for(int i=0; i<n; ++k; ++i) {
```

```
for int (k=0; k<n; ++k){
    while (t[k] > t[k+1]) {
        swap(t[k], t(k+1]);
        k:0;
    }
}
```

$x > y$ et $3 < y$

$x, y, 3$ swap(x,y)

$x, y, 3 \wedge x > y \wedge 3 < y \Longrightarrow y, x, 3 \wedge x > y \wedge 3 < y$

Γ 3 1

si $x > y$ et $y > 3$

on $x > 3$

donc swap(x,3);

$y, 3, x \Rightarrow$ swap(y,3).

preuve de la terminaison :

effet la sequence correspond a des echanges tant qu'une condition est vérifiée.

l'algo sait quand le tri est fini donc quand toutes les valeurs sont croissantes

x_1, x_2, x_3

$k=0$ $x_1=5$ $x_2=3$ $x_3=1$

↓ swap(x_1, x_2)

$k=0$ $x_1=3$ $x_2=5$ $x_3=1$

↓ swap(x_2, x_3)

$k=0$ $x_1=3$ $x_2=1$ $x_3=5$

↓ swap(x_1, x_2)

$x_1=1$ $x_2=3$ $x_3=5$

$$y \geq x \xrightarrow{x=0} y \geq 0$$

12) $y = 0 \xrightarrow{y \leftarrow y+1} y = 1$

13) $y = x - 1 \xrightarrow{y \leftarrow y+1} x = y$

14) $y = x - 2 \xrightarrow{y \leftarrow y+1} x = y + 1$

$y = x - 1$

15) $y < x \xrightarrow{y \leftarrow y+k} y \leq x$

$\| \; y > x$

Preuve de la correction partielle:

Soit pos 1, pos 2, pos 3 des positions successives;

on pose : pos 1 = a, pos 2 = b, pos 3 = c

Soit : $a > b \wedge b > c$

donc : $a > c$

$\boxed{pos\,1 = a \wedge pos\,2 = b} \wedge pos\,3 = c$

$\Downarrow t \leftarrow pos\,1$

$pos\,2 = b \wedge t = a$

$\Downarrow pos\,1 \leftarrow pos\,2$

$pos\,1 = b \wedge t = a$

$\Downarrow pos\,2 \leftarrow t$

$pos\,1 = b \wedge \boxed{pos\,2 = a \wedge pos\,3 = c}$

$\Downarrow t \leftarrow pos\,2$

$t = a \wedge p_3 = c$

$\Downarrow p_2 \in p_3$

$t = a \wedge p_2 = c$

$\Downarrow p_3 \in t$

$p_1 = b \wedge p_2 = c \wedge p_3 = a$

2.3, 2.4.2.5.26.2.7.2.8.

$k = n_4 - 1$

$f_r n_1;$

tanq que $k \neq 0;$

$f x = k$

$-- k_i;$

$n \times 2 \times 3 \times 4 \times 5$

$k-2 \quad k-1 \quad k$

5×4

E.2.1 :

```
F ← n !
k ← n-1 4
   invariant de boucle  0 ≤ k ≤ n
   qté de calcule : k
Tant que  k > 0
   F ← F x k
      k ← k - 1
```

assertion d'exactitude

AS : F = n !

Séance de TD1 : Pr Haneart

Exo chap2 :

Exo 2.1

1) $x = 1 \wedge y = 2 \xrightarrow{y \leftarrow y+1} x = 1 \wedge y = 3$

2) $x = 1 \wedge y = 2 \xrightarrow{y \leftarrow y+x} x = 1 \wedge y = 7$

3) $x = y+1 \xrightarrow{y \leftarrow y+1} y = x$

4) $x = y \xrightarrow{y \leftarrow y+1} x = y-1$ ou $y = x+1$

5) $x = y \xrightarrow{y \leftarrow y-1} y = x-1$

6) $x \neq 0 \wedge y$ est un multiple de $x \xrightarrow{y \leftarrow y-x}$

$x \neq 0$ et y un multiple de x.

7) $y = kx \xrightarrow{y \leftarrow y+x} y = kx + x = (k+1)x$.

8) $y = (k+1)x \xrightarrow{k \leftarrow k+1} y = kx$.

9) $y = (k+1)^2 x \xrightarrow{k \leftarrow k+1} y = k^2 x$.

10) $y = k^2 x \xrightarrow{k \leftarrow k+1} y = (k+1)^2 x$.

$y = k^2 x \xrightarrow{k \leftarrow k+1} y = (k^2 - 1)x$

11) $y = k^2 x \xrightarrow{x \leftarrow 2x} y = k x$.

12)
$$y \leftarrow y+1$$
$$y = 0 \Longrightarrow y = 1$$

13)
$$y = x - 1 \xrightarrow{y \leftarrow y+1} x = y$$

14)
$$y \leftarrow y+1$$
$$y = x - 2 \Longrightarrow x = y + 1$$

$y = x - 1$

17) $y + k \leqslant x \xrightarrow{y \leftarrow y+k} y \leqslant x$

16) aucune $\xrightarrow{y \leftarrow 0} y \geqslant 0$

hypothèse.

- Rien.

2.3) Preuve de la correction partielle:

soit x, y, z 3 objets respectivement croissant de taille

k ← 0

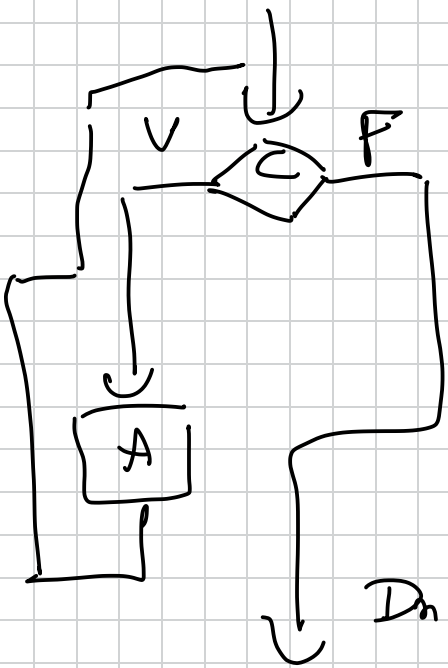
x : a, y : b, z = c

tant que k < nv Fait

tant que T[k] > T[k+1] faire

~~///~~ trier(α, β)

k ← k + 1.



on a : C ≡ ∃ un objet x tel que, ∃ y un objet y situé plus loin, tel que taille(x) > taille(y)

Donc : ¬C ≡ ∀ un objet x

∀ y tel qu'il soit situé après x

on a taille(x) ≤ taille(y)

Donc : ¬C ≡ ∀ un objet x, il n'y a pas

y qui est initialement situé après

et tel que taille(x) < taille(y).

Donc la ribambelle est ↗.

2.4.

	I B	Q C
1)	$0 \leqslant k \leqslant n$	
2)	$1 < k \leqslant n+2$	
3)	$0 \leqslant k \leqslant n$	
4)	$0 < k \leqslant n$	
5)	$0 < k \leqslant n$	
6)	$k < n$ et $\neg B$	

par contre si $n < 0$ toutes les boucles χ terminent.

on a supposé $n \geqslant 0$.

1) $0 \leqslant k \leqslant n$

$1 \leqslant k \leqslant n+1$

Q : $1 \leqslant k \leqslant n+1$

c : $k \leqslant n$ Q ∧ c :

$1 \leqslant k \leqslant n+1 \wedge k \leqslant n$

$1 \leqslant k \leqslant n$

$\xrightarrow{k+1 \leq n} 2 \leq k \leq n+1$

$0 \leq k \leq n$

$1 \leq k \leq n+1$

$0 \leq \ell \leq n-1$

$0 \leq \ell \leq n$

P pour

je m'appelle P

$0 < k \leq n$

Alexandre

$1 \leq k \leq n$

je m'appelle

pour.

je m'appelle

$0 \leq k < n$

pour

φ: $0 \leq k \leq n+1$

je m'appelle

pour

c: $k < n$

$\Rightarrow 0 \leq k < n$

$\xrightarrow{k+1} 1 \leq k < n+1$

$k = n$ $\quad k > 0$

$0 < k \leq n$

F $\quad 1 \leq k \leq n$

$0 \leq k \quad \wedge \quad k > 0$

$0 < k$

$1 \leq k$

6)

Rappels: Logique de Hoare

But. Construire des programmes corrects tout en prouvant qu'ils le sont.

Correction: Correction Partielle + Terminaison

Invariant de Boucle: Expression qui dépend de variables (assertion) qui ne varient pas au long de la boucle.

quantité de Contrôle:

Expression qui dépend des variables:
- A valeurs entière
- Strictement monotone (incrémente ou décrémente)
- Bornée.

Exercice 2.2:

1) $x=5 \wedge y=2 \xrightarrow{y \leftarrow y+1} x=5 \wedge y=3$

2) $x=5 \wedge y=2 \xrightarrow{y \leftarrow y+x} x=5 \wedge y=7$

3) $x=y+1 \xrightarrow{y \leftarrow y+1} y=x$

4) $x=y \xrightarrow{y \leftarrow y+1} y=x+1$

5) $x=y \xrightarrow{y \leftarrow y-1} y=x-1$

6)

7)

8)

9)

10)

11)

12)

13)

14)

15)

16)

Exercice 2.3 : P = la ribambelle d'objets contient n objets desordonnés / Q = la ribambelle contient n objets ordonnés.

Exercice 2.4 :

1) $k \leftarrow 0$
tant que $k < n$ Faire
$k \leftarrow k + 1$
A

IB : $0 \leq k \leq n$

Qu Cont : $k \in \mathbb{N}$

2) $k \leftarrow 1$

tant que $k \leq n$ faire

$k \leftarrow k+1$

A

IB: $1 \leq k \leq n+1$

φ_c : k

3) $k \leftarrow 0$

tant que $k < n$ faire

A

$k \leftarrow k+1$

IB: $0 \leq k \leq n$

φ_c : k.

4) $k \leftarrow n$

tant que $k > 0$ faire

$k \leftarrow k-1$

A

IB: $0 \leq k \leq n$

φ_c : k.

5) $k \leftarrow n$

tant que $k > 0$ faire

A

$k \leftarrow \lfloor \frac{k}{2} \rfloor$

IB: $0 \leq k \leq n$

φ_c : k

6) $b \leftarrow$ FAUX

$k \leftarrow 0$

tant-que $\neg b \wedge k < n$ faire

si c alors

A

$b \leftarrow$ Vrai

sinon

A'

$k \leftarrow k + 1$

$(0 \leq k \leq n)$ et $(\neg b = \text{Vrai})$

Exercice 2.5

$k \leftarrow n$

$f \leftarrow 1$

tant que $k > 0$ Faire

$f \leftarrow f \times k$

$k \leftarrow k - 1$

IB : $0 \leq k \leq n$

φc : k.

IB : se termine. (preuve de terminaison).

φ : $0 \leq k \leq n \wedge f = k!$

c : $k > 0$

$\neg c$: $k \leq 0$

$k = 0$ et $k \leq n$

$k = n \wedge f = k!$

Donc : $f = n!$

Exercice 2.6 :

$k \leftarrow 1$
somme $\leftarrow 0$
tant que $k <= n$ faire
Somme $\leftarrow$ somme $+ \frac{1}{k}$
$k \leftarrow k+1$

$1 \leq k \leq n+1$

$k \leftarrow n$
Som $\leftarrow 0$
tant que $k >= 1$ faire
Some $\leftarrow$ Some $+ \frac{1}{k}$
$k \leftarrow k-1$

Exercice :
Données : n (entier)
à faire : écrire la suite des valeurs entières en décr.
-oissant de n à 1.

φ : $0 \leq k \leq n \wedge$ les valeurs entières de n à k+1 ont été affichées en y. (α)

c : $k > 0$

$\varphi \wedge c \Rightarrow 0 \leq k \leq n \wedge \alpha \wedge k > 0$

$\varphi \wedge c \Rightarrow 1 \leq k \leq n \wedge \alpha$.

écrire k
$\Rightarrow$ —— $\wedge \alpha$ ou à k

k←k-1
$\Rightarrow 0 \leq k \leq n-1 \wedge$ —— ou à k+1
$\Rightarrow \varphi$

φ : $0 \leqslant k \leqslant n$

c : $k < n$

$\varphi \wedge c$: $0 \leqslant k \leqslant n \wedge k < n$

: $0 \leqslant k < n$

$\xrightarrow{k \leftarrow k+1}$ $1 \leqslant k \leqslant n$

$\varphi \wedge \neg c$: $0 \leqslant k \leqslant n \wedge k \geqslant n$

Preuve de terminaison $\boxed{k = n}$

φ : $1 \leqslant k \leqslant n+1$

c : $k \leqslant n$

$\varphi \wedge c$: $1 \leqslant k \leqslant n+1 \wedge k \leqslant n$

$1 \leqslant k \leqslant n$

$\xrightarrow{k \leftarrow k+1}$ $2 \leqslant k \leqslant n+1$

$\Longrightarrow \varphi$

$\varphi \wedge \neg c$: $1 \leqslant k \leqslant n+1 \wedge k > n$

Donc : $k = n+1$

Preuve de la terminaison du programme.

Exercice 2.6:

Données: n, entier naturel
résultat: $h = H_n = R$.

φ : $0 < k \leq n \wedge h = H_k$ $\qquad H_n = \sum_{k=1}^{n} \frac{1}{k}$ n naturel

c : $k < n$

φ et $\neg c \Rightarrow$ $0 < k \leq n \wedge h = H_k \wedge k \geq n$

$\Rightarrow$ $k = n \wedge h = H_k$

$\Rightarrow$ $h = H_n$.

$\varphi \wedge c \Rightarrow$ $0 < k \leq n \wedge h = H_k \wedge k < n$

$\Rightarrow 0 < k \leq n-1 \wedge h = H_k$

$\overset{h \leftarrow h + \frac{1}{k+1}}{\Rightarrow} 1 \leq k \leq n \wedge h = H_{k-1}$

$\overset{k \leftarrow k+1}{\overset{h \leftarrow h + \frac{1}{k}}{\Rightarrow}} 1 \leq k \leq n \wedge h = H_k$

$\Rightarrow \varphi$.

Algorithme qui calcule une somme de 1 à n

Algorithmique 1 : méthodologie de la programmation impérative
Fiche d'aide à la construction d'algorithmes de calculs simples (A'' vide)

NOM PRÉNOM : ZENTAR Ghiles

0) Expression dont l'algorithme de calcul est développé par la suite

$s \leftarrow 0$
$k \leftarrow 1$
tant que $k \leq n$ faire : $s \leftarrow s + k$ $k \leftarrow k + 1$

1) Assertion d'entrée P

Précondition aucune

2) Assertion de sortie R

$$s = \sum_{k=1}^{n} k = \frac{k(k+1)}{2}$$

3) Assertion Q (futur invariant de boucle)

$$1 \leqslant k \leqslant n+1 \wedge s = \frac{k(k+1)}{2}$$

4) Fin de l'induction : expression d'une condition (d'itération) C telle que $Q \wedge \neg C \Rightarrow R$

$C : k \leqslant n$ $\neg C : k > n$ $Q \wedge \neg C : 1 \leqslant k \leqslant n+1 \wedge s = \frac{k(k+1)}{2} \wedge k > n$
$\Rightarrow k = n+1 \wedge s = \frac{k(k+1)}{2}$ } on sommes puis on incrémente. Donc la boucle prend fin quand $k = n+1$

5) Preuve que $Q \wedge \neg C \Rightarrow R$

$Q \wedge \neg C \Rightarrow 1 \leqslant k \leqslant n+1 \wedge s = \frac{k(k+1)}{2} \wedge k > n$
$\Rightarrow k = n+1 \wedge s = \frac{k(k+1)}{2}$] k pas encore incrémenté donc $k = n$
$\Rightarrow s = \frac{k(k+1)}{2}$] $\Rightarrow s = \frac{n(n+1)}{2}$

7) Induction (hérédité) : expression d'une instruction A' qui se termine et qui satisfait le schéma pré-post $Q \wedge C \overset{A'}{\Longrightarrow} Q$

$1 \leqslant k \leqslant n+1 \wedge s = \frac{k(k+1)}{2} \wedge k \leqslant n$ $k < n+1$
$\Rightarrow 1 \leqslant k \leqslant n \wedge 1 = \frac{k(k+1)}{2}$
$k \leftarrow k+1$
$\Rightarrow 2 \leqslant k \leqslant n+1, 1 \; s = \frac{k(k+1)}{2}$
C'est donc à l'itération de l'IBdm satisfait $Q \wedge C \overset{A'}{\Longrightarrow} Q$.

6) Preuve que $Q \wedge C \overset{A'}{\Longrightarrow} Q$

$1 \leq k \leq n+1 \wedge s = \frac{k(k-1)}{2} \wedge k < n$

$\Longrightarrow 1 \leq k \leq n \wedge s = \frac{k(k+1)}{2}$

$\Longrightarrow k \leq n \wedge s = \frac{k(k+1)}{2}$

$\Longrightarrow k = n \wedge s = \frac{k(k+1)}{2}$

$\Longrightarrow s = \frac{n(n+1)}{2} \wedge k = n$

8) Quantité de contrôle pour la boucle associée au schéma pré-post $Q \wedge C \overset{A'}{\Longrightarrow} Q$

le qtté de ctrl est k.

9) Base (initialisation) : expression d'une instruction A qui se termine et qui satisfait le schéma pré-post $P \overset{A}{\Longrightarrow} Q$

A) C : déclaration des paramètres

B) C : déclaration et initialisation des autres variables locales

```
int k = 1
int s = 0
```

C) C : expression de la boucle

```
int k = 1;
int s = 0;
while (k <= n) {
s = s + k;
++k;
}
```

Calcul de la factorielle :

P (Précondition d'entrée : aucune).

R (Assertion de sortie). R : $f = n!$.

Ennoncé du Futur IB :

Φ : $0 \leqslant k \leqslant n \wedge f = k!$

C : $k < n$

Alors : $\Phi \wedge \neg C \Rightarrow R$

puisque :

$$\Phi \wedge \neg C \Rightarrow 0 \leqslant k \leqslant n \wedge f = k! \wedge k \geqslant n$$
$$\Rightarrow k = n \wedge f = k!$$
$$\Rightarrow f = n!$$

$\Phi \wedge C \Rightarrow \Phi$

$\Phi \wedge C \Rightarrow 0 \leqslant k \leqslant n \wedge f = k! \wedge k < n$

$\Rightarrow 0 \leqslant k \leqslant n-1 \wedge f = k!$

$\xrightarrow{k \leftarrow k+1} 1 \leqslant k \leqslant n \wedge f = (k-1)!$

$\xrightarrow{f \leftarrow f \times k} 1 \leqslant k \leqslant n \wedge f = (k-1)! \times k = k!$

Exercice 2.5 : Reprenter factorielle en décroissant.

Φ : $0 \leqslant k \leqslant n \wedge f = k!$

C : $k > 0$

$\Phi \wedge C \Rightarrow 0 \leqslant k \leqslant n \wedge f = k! \wedge k > 0$

$\Rightarrow 1 \leqslant k \leqslant n \wedge f = k!$

$\xrightarrow{k \leftarrow k-1} k \leqslant n-1 \wedge f = (k+1)!$

$\xrightarrow{f \leftarrow f \times k} k \leqslant n-1 \wedge f = (k+1)! \times k]$

$\Rightarrow k \leqslant n-1 \wedge f = k!$

$\Phi \wedge \neg C \Longrightarrow 0 \leqslant k \leqslant n \wedge f = k! \wedge k \leqslant 0$

$\Longrightarrow k = n$ et $f = n!$ et $f = k!$

$\overset{\text{Analyse}}{\Longrightarrow} k = 0$ et $f = n!$.

Exercice 2.6 :

$$H_n = \sum_{k=1}^{n} \frac{1}{k}.$$

$\Phi : 0 \leqslant k \leqslant n+1 \wedge h = H_k$

$C : k < n+1$

$\Phi \wedge C \Longrightarrow 0 \leqslant k \leqslant n+1 \wedge h = H_k \wedge k < n+1$

$\Longrightarrow 0 \leqslant k \leqslant n \wedge h = H_k$

$\overset{k < k+1}{\Longrightarrow} 1 \leqslant k \leqslant n+1 \wedge h = H_{k-1}$

$\overset{n < n+1}{\Longrightarrow} 1 \leqslant k \leqslant n+1 \wedge h = H_{k-1} + \frac{1}{k} = H_k$

$\Phi \wedge \neg C \Longrightarrow 0 \leqslant k \leqslant n+1 \wedge h = H_k \wedge k \geqslant n+1$

$\Longrightarrow k = n+1 \wedge h = H_n$

Exercice 2.9 :

Donnée : x

$S : \sum_{j=0}^{n} \frac{x^{2j}}{(2j+1)!}$

$\varphi : 0 \leq k \leq n \quad \wedge \quad S = \sum_{j=0}^{k} \frac{x^{2j}}{(2j+1)!} \quad \wedge \quad t = \frac{x^{2k}}{(2k+1)!}$

$c : k < n$

$\varphi \wedge \neg c \Longrightarrow 0 \leq k \leq n \wedge \varphi \wedge k \geq n$

$\Longrightarrow k = n \wedge S = \sum_{j=0}^{n} \frac{x^{2j}}{(2j+1)!}$

$\Longrightarrow S = \sum_{j=0}^{n} \frac{x^{2j}}{(2j+1)!}$

$\varphi \wedge c \Longrightarrow 0 \leq k \leq n \wedge k < n \wedge \varphi$

$\Longrightarrow 0 \leq k < n \wedge \varphi$

$k+1 \leq$

$\Longrightarrow 1 \leq k \leq n \wedge S = \sum_{j=0}^{k-1} \frac{x^{2j}}{(2j+1)!} \wedge$

$t \leftarrow t \times x^2 / ((2k)(2k+1)) \quad \Longrightarrow \quad \_\_\_\_ \wedge t = \frac{x^{2k-2}}{(k-1)!}$

$\_\_\_\_ \wedge t = \frac{x^{2k}}{(2k+1)!}$

$S \leftarrow S + t$

$\Longrightarrow \_\_\_\_ \wedge \_\_\_\_ \wedge S = \sum_{j=0}^{k} \frac{x^{2j}}{(2j+1)!}$

$\Longrightarrow \varphi.$

?

$k = 0$

$t = 1$

$S = 1$

Language C :

```
int k=0;
double t=1.0;
double s=1.0;
while (k<n){
  ++k;
  t*= x*x/(2*k)/(2*k+1);
  s+=t;
```

TP 3 brouillon.

Exercice 1 :

$\sum_{j=0}^{n} \frac{x^{2j+1}}{(2j+1)!}$ } faire la fonction puissance et " " factorielle

<u>fct puissance</u> :

l'expression c'est x^{2j+1}

IB : $1 \leq k \leq 2j+2 \wedge p = x^{2j+1}$

C : $k \leq 2j+1$

$n = 2j+1$ $\quad x^3$

x^n $\quad x \times x \times x$

$\underbrace{\quad\quad\quad\quad}_{n-1 \text{ fois}}$

$x \times x \times x \times x - - - - \times x$

```
while (k ≤ n) {
    p = p * x
    + + k;
```

$Q \wedge c : 1 \leq k \leq n+1 \wedge p = x^k$

$: k \leq n$

$\Longrightarrow 0 \leq k \leq n \wedge p = x^k$

$k \leftarrow k+1$

$\Longrightarrow 1 \leq k \leq n+1 \wedge f = x^{k-1}$

$f \leftarrow f \times x$

$\Longrightarrow 1 \leq k \leq n+1 \wedge f = x^{k-1} \times x = x^k$

$\Longrightarrow 0 \leq k \leq n+1 \wedge f = x^k.$

$Q \wedge \neg c : 0 \leq k \leq n+1 \wedge p = x^k$

$\wedge k > n$

$\varphi \wedge \neg c$: $k \leq n+1 \wedge p = x^k \wedge k \geq n+1$

$\varphi \wedge \neg c$: $k = n+1 \wedge p = x^{k+1}$

x^r

$k=1$ $\qquad$ $k=r$

$x \quad x$

x^{2r+1}

$(2r+1 \not\equiv 2! = 0)$

$\varphi = x$ $\qquad$ x^2 $\qquad$ $x \cdot x = \frac{2-1}{2}$

$x \cdot x \quad x \quad x \cdot x \quad x \quad x \cdot x \quad x \cdot x$

φ: $0 \leq k \leq 2r+2 \wedge x^k$

c: $k \leq 2r+1$

$k \leftarrow k+1$

$\Rightarrow 1 \leq k \leq 2r+2 \wedge x^{k-1}$

$p := p \cdot x$ $\Rightarrow 1 \leq k \leq 2r+2 \wedge x^k$

$\Rightarrow \varphi$

φ: $0 \leq k \leq n+1 \wedge f = k!$

c: $k \leq n$

$k \leftarrow k+1$

$\Rightarrow 1 \leq k \leq n \wedge f = (k-1)!$

x^{2r+1}

$1 \leq k \leq 2r+2$ $\qquad$ x^k

$k \leq 2r+1$

x^{2j+1}

$x^{1 \text{ ou } 3 \text{ ou } 5 \text{ ou } 7}$

$(2j+1)!$ $(1 \text{ ou } 3 \text{ ou } 5 \text{ ou } 7)!$

$\frac{3^5}{5!}$

$1 \times 2 \times 3 \times 4 \times 5$

$\frac{3 \times 3 \times 3 \times 3 \times 3}{1 \times 2 \times 3 \times 4 \times 5} = \frac{81}{40} = 2$

$= 2,025$

Q : $0 \leq j \leq n+1$ $\sum_{j=0}^{n} \frac{x^{2j+1}}{(2j+1)!}$

C : $j \leq n$.

$\frac{x^1}{1!} = x^0$

$\frac{x^2}{(2j \times 2j+1)}$ 1

$\frac{x^3}{6}$

$(2 \times 2 + 1)$

$3!$

$1 \times 2 \times 3 \times 4 \times 5$

$6 \times 4 \times 5$

$\frac{x^5}{(2j \times 2j+3)}$

$\frac{x^5}{6 \times 4 \times 5}$

Exercice n 2,9.2 :

$$\sum_{j=0}^{n} (-1)^j \frac{x^j}{j!}$$

Données :
x et n

résultat $s = \Sigma$

Q : $0 \leq k \leq n \wedge s = \sum_{j=0}^{k}$

C : $k < n$

$Q \wedge C$: $0 \leq k \leq n \wedge s = \sum_{j=0}^{k} \wedge k < n \wedge t = -1^k \frac{x^k}{k!}$

$Q \wedge C$: $0 \leq k \leq n-1 \wedge s = \Sigma \wedge t$

$Q \wedge C \xrightarrow[x-s+t]{k=k+1} 1 \leq k \leq n \wedge s = \sum_{j=0}^{n-1} \wedge t$

$Q \wedge C \Longrightarrow 1 \leq k \leq n \wedge s = \sum_{j=0}^{n} \wedge t$

$\Longrightarrow Q$.

$Q \wedge \neg C$: $0 \leq k \leq n \wedge s = \sum_{j=0}^{k} \wedge k \geq n \wedge t$

$: k = n \wedge s = \sum_{j=0}^{k} \wedge t$

$\Longrightarrow s = \sum_{j=0}^{n}$.

1) IB: $0 \leq k \leq n \wedge s = \sum_{j=0}^{k} x^j \wedge t = x^k$

2.8.1:

$(k+1)^2 = k^2 + k + k + 1$

$\varphi : 0 \leq k \leq n \wedge A_{i,n} = (k+1)^2$

$c : k < n$

Donc k < n.

$\varphi \wedge c : 0 \leq k \leq n \wedge A_{i,n} = (k+1)^2 \wedge k < n$

$\varphi \wedge c : 0 \leq k \leq n-1 \wedge A_{i,n} = (k+1)^2$

$\varphi \wedge c \xRightarrow{k \leftarrow k+1} 1 \leq k \leq n \wedge A_{i,n} = ((k-1)+1)^2$

$\varphi \wedge c \xRightarrow{A \leftarrow A}$

$0 \quad 1 \quad 2$

$0^2 \quad 1^2 \quad 2^2$

$2 \times 2 \times 1 = 4$

3^2

4^2

k^2

(1) $k^2 \xRightarrow{2k+1} (k+1)^2$ 2

$(k+1)^2 \xRightarrow{2k+j} (k+2)^2$ 3

$2k + j + 1$

$k^2 + 4k + 4$

$(k+3)^2 = k^2 + 6k + 9$

$$(k+1)^3 = k^3 + k^2 + k^2 + k + k + k + 1$$

$$1 = k^2$$

$$S = k^3$$

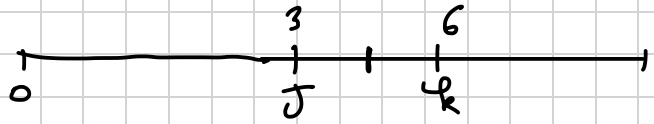
Exo 2.11 :

Φ : $\text{Pgcd}(P, q) = \text{Pgcd}(a, b)$

C : $P \neq q$.

$$\frac{J + k}{2}$$

```
int Div = J + k ;
    Div /= 2;
```



$$\frac{6+3}{2} = 4,5$$

$$J + (k - J)/2 = 4,5$$

$k \geq 1 \wedge S = \sum_{j=1}^{k-1} \frac{1}{j} \Longrightarrow k \geq 1 \wedge \sum_{j=1}^{k} \frac{1}{j}$

$S = \sum_{j=1}^{k-1} \frac{1}{j} = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k-1}$

Si on ajoute à S, $\frac{1}{k}$:

$S = \frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{k-1} + \frac{1}{k}$

$S = \sum_{j=1}^{k} \frac{1}{j}$

$k \geq 0 \wedge 1 = \frac{(2k+1)!}{4^k ((k+1)!)^2} 2^{2k+1} \Longrightarrow$ $k \leftarrow k+1$

$k \geq 1 \wedge 1 = \frac{(2(k-1)+1)!}{4^{k-1} ((k-1+1)!)^2} 2^{2(k-1)+1}$

$1 : \frac{(2k-1)!}{4^{k-1} (k!)^2} 2^{2k-1}$ $\quad 0 \leq k \leq \frac{n}{2} + 1$

$\left\lceil \frac{k}{2} - 1 \right\rceil$

① $n = 0$

$\Rightarrow \left\lceil \frac{0}{2} + 1 \right\rceil = 1$

$Q : 0 \leq k \leq n+1$

② $n = 1$

$\left\lceil \frac{1}{2} + 1 \right\rceil = 2 : k \leq n$

1 2 3

0 1 $k = n+1$

-2 -1

2 4

1 $\frac{1}{2} + 1$.

$\frac{2}{2} + 1$ n pair $\quad -2 \leq k \leq n$ $\quad \frac{0}{2} + 1$

$n+1$ impair $\quad -1 \leq k \leq n$. $\quad 2$.

$n = 3 \quad \left\lceil \frac{3}{2} + 1 \right\rceil = 3$

$n = 4 \quad = 3$

$n = 5 \quad = 4$

$n = 6 \quad = 4$

$n = 7 \quad = 5$

$n = 8 \quad = 5$

1) Pour coder un entier strictement négatif si besoin en cas d'erreur

2) Cours page 38

3) long int, Pour des deplacements et taille de fichiers

4) 4 → A
3 → C
2 → B

4) appel binaire update
1 → D.

1. appel
2. open binaire
3. appel update

"x" Creer.
"r" lire
"w" écrire.
"+" update.

11 → F
12 → H
13 → E
14 → G
9 → P
10 → N
13 → O
14 → M

1 → D
2 → B
3 → C
4 → A
5 → K
6 → I
7 → L
8 → J

w+ tronque vider puis pour lecture écriture
r+ on ne commence pas par vider le fichier.

5.3.
```
bool fexists(const char *filename){
    FILE *f = fopen(filename, "r");
    if(f == NULL){
        return false;
    }
    fclose(f);
    return true;
}
```

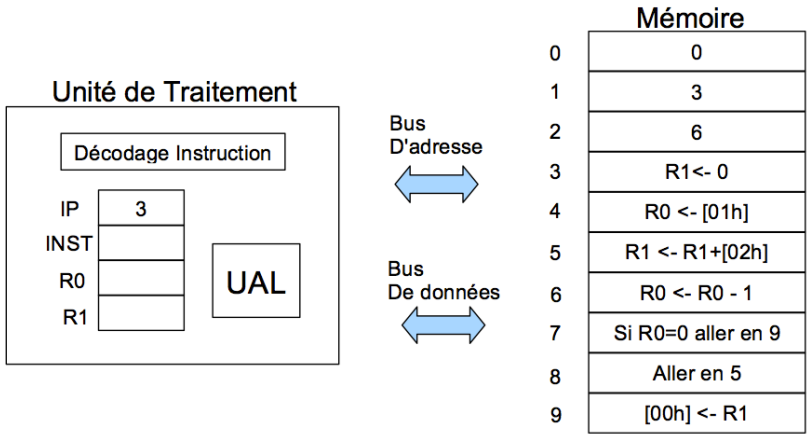
5.4.
```
long int fnlines(const char *filename){
    FILE *f = fopen(filename, "r");
    if(f == NULL){
        return -1;
    }
    long int n = 0;
    int c = fgetc(f);
    while(c != EOF){
        n += (c == '\n');
        c = fgetc(f);
    // Qc : nombre d'appel fgetc(f).
    // IB : Dernier caractère renvoyé par l'appel fgetc(f)
    //      && n == nb de '\n' dans c si c != EOF
    if(feof(f) != 0){
        n = -1;
    }
    if(fclose(file) != 0){
        n = -1;
    }
    return n;
}
```

5.5. 1)

```
int fcopy(const char * destfn, const char
                 * srcfn) {
    FILE * fsrc = fopen(fsrc, "r");
    if (fsrc == NULL) {
        return -1;
    }
    FILE * fdest = fopen(fsrc, "wx");
    if (fdest == NULL) {
        fclose(fsrc);
        return -1;
    }
    int c; // voir pas 44.
    // sans tenir compte des effets de l'ELB
    // IB : tous les caractères lus si fsrc          ont été écrits
    // Qc : nb après fgetc(fsrc)                     dans fdest
    int r = 0;
    if (!feof(fsrc)) {
        r = -1;
    }
    if (fclose(fsrc) != 0) {
        r = -1;
    }
    if (fclose(fdest) != 0) {
        r = -1;
    }
    return r;
}
```

TD n°4

Exercice 1



Soit un ordinateur simplifié avec l'architecture présenté sur la figure ci-dessus. À l'initialisation de la machine, le registre IP (compteur ordinal) contient la valeur 3.

1- En supposant chaque cellule de la mémoire RAM est de taille 1 octet, donner la taille du segment de données et la taille du segment de code.
2- Exécuter pas à pas les instructions et donner à chaque pas le contenu des registres IP, INST, R0 et R1.
3- Combien de lectures et d'écritures en mémoires sont effectuées par ce programme ?
4- Quel est le rôle du compteur ordinal et le rôle du registre d'instructions INST ?
Que fait ce programme ?

Exercice 2

Traduire en langage machine l'extrait de programme assembleur 8086 suivant en vous servant de la table de correspondance codop/instructions ci-jointe. On supposera par ailleurs que CS = 48F1h, que l'étiquette « data » réfère au numéro de segment 48DFh, et que l'étiquette « TAB » est associée à une variable se trouvant dans le segment de données avec une valeur de déplacement qui est 001Bh.

```
debut :    mov ax,data
           mov ds,ax
           mov AX, DS :[0002]
           mov CX, AX
           lea BX, TAB
           xor SI,SI
boucle :   mul CL
           add SI, 2
           mov [BX+SI], AX
           loop boucle
           mov AX, 4C00h
           int 21h
```

Table de correspondance code opérations/instructions :

Symbole	Code Op.	Octets	Symbole	Code Op	Octets
MOV AX, valeur	B8	3	DEC CL	FE C9	2
MOV BL, valeur	B3	2	MUL CL	F6 E1	2
MOV AX, DS:[adresse]	A1	3	ADD SI, valeur	83 C6	3
MOV AX, BX	8B C3	2	ADD AX, valeur	05	3
MOV CX, AX	8B C8	2	ADD AX, DI	03 C7	2
MOV CX, DS:[adresse]	8B 0E	4	JNE adresse relative	75	2
MOV CL, DS:[adresse]	8A 0E	4	JNC adresse relative	73	2
MOV DS, AX	8E D8	2	LOOP adresse relative	E2	2
MOV DI, DS:[adresse]	8B 3E	4	MOV SI, valeur	BE	3
LEA BX, adresse	BB	3	CMP CL, valeur	80 F9	3
XOR BX, BX	33 DB	2	MOV [SI],AX	89 04	2
XOR SI, SI	33 F6	2	MOV [BX+SI], AX	89 00	2
INC BX	43	1	INT valeur	CD	2

Exercice 3

On suppose pour la suite que le segment de données contient les informations suivantes :

DS :0000 à DS :000F 04 52 04 00 12 98 01 01 01 01 01 01 01 01 01 01
DS :0010 à DS :001F E9 EF FF 21 A6 B4 D8 FA 4F 5E A5 36 DF B1 C9 A1

Voici un extrait de programme, implanté dans le segment de code à CS :0005h.

Code instruction	Opération
A1 02 00	MOV AX,DS:[0002]
8A C8	MOV CL,AL
BE 00 00	MOV SI,0
FE C9	boucle : DEC CL
F6 E1	MUL CL
83 C6 02	ADD SI,2
89 04	MOV [SI],AX
80 F9 00	CMP CL,1
75 F2	JNE boucle

1. Donnez une structure de programme assembleur 8086 (juste la déclaration du segment de données) ayant permis d'initialiser les 16 premiers octets du segment de données ci-dessus.

2. Complétez sur votre feuille le tableau ci-dessous en donnant l'état des registres après exécution des différentes instructions :

adresse	Opération	IP	SI	AX	CX
	état initial		0000	0000	0000

calcul factorielle 4! = 4x3x2x1

3. Expliquez ce que fait la suite d'instructions précédente. Quel est le rôle du registre CL ?

Photo

4. Donnez le contenu de la mémoire après exécution de cet extrait de programme.

5. Combien de lectures et d'écritures en mémoire centrale sont effectuées par cet extrait de programme?

Ecritures (sockets), lecture de données DS:[0002], 21 lectures d'instr.
22 au total

6. Justifiez la valeur d'opérande F2 dans le codage en langage machine de l'instruction de l'instruction « JNE boucle » (dernière ligne du programme ci-dessus)

F2 code -14 saut -14 octets à la place 001BH 0001

+14 : 0000 0111₂

1111 0001

1111 0010

F 2

Chap 4 Algo :

Tableaux = suite finie de variables.

a = = &a [0].

```
struct tm {
int tm_hour;
int tm_min;
int tm_sec;
}
```

```
*P -> tm_hour = = 0;
*P -> tm_min = = 0;
*P -> tm_sec = = 0;
```

```
FILE *fp = fopen(file, "rb");
if(fp == NULL){
    return -1;
}
size_t j = 0;
elemt k;
size_t n = fread(&k, sizeof(elemt), 1, fp);
while(k <= ap){
}
++j;

if(!feof(fp)){
        n = -1; };
if(fclose(fp) != 0){
    n = -1;
}
    return n;
}
```

Fonctions usuelles :

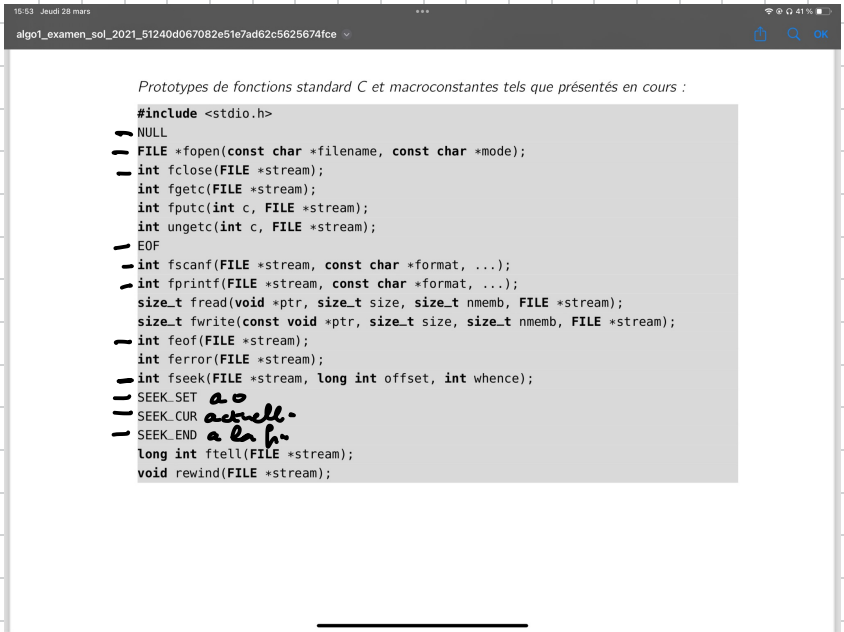
fseek :

```
FILE *file;
file = fopen("file.txt", "w");
if (file == NULL){
    printf("erreur\n");
    return -1;
}
fprintf(file, "0123456789\n");
fseek(file, 0, SEEK_SET);
```

SEEK_SET à 0

SEEK_CUR actuelle

SEEK_END at end of file



algo1_examen_sol_2021_51240d067082e51e7ad62c5625674fce

Prototypes de fonctions standard C et macroconstantes tels que présentés en cours :

```
#include <stdio.h>
NULL
FILE *fopen(const char *filename, const char *mode);
int fclose(FILE *stream);
int fgetc(FILE *stream);
int fputc(int c, FILE *stream);
int ungetc(int c, FILE *stream);
EOF
int fscanf(FILE *stream, const char *format, ...);
int fprintf(FILE *stream, const char *format, ...);
size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);
size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE *stream);
int feof(FILE *stream);
int ferror(FILE *stream);
int fseek(FILE *stream, long int offset, int whence);
SEEK_SET à 0
SEEK_CUR actuelle
SEEK_END à la fin
long int ftell(FILE *stream);
void rewind(FILE *stream);
```

Fprintf :

```
FILE *fichier = fopen("fichier.txt", "w");
  if (fichier == NULL){
        return -1;
      }
  fprintf(file, "\n");

  fprintf(file, "\t-ABC\n");
  fclose(file);

    fscanf(file, "format String", ----)
```

les structures :

types de variables.

livre : titre, auteur, date

```
struct livre {
    char titre[100];
    char auteur[100];
    int annee;
};
```

```c
long int flinear_right_search(FILE *stream, size_t size, const void *ptr,
                              int (*cmp)(const void *x, const void *y)){
    if(fseek(stream, 0, SEEK_END) != 0){
        return -1;
    }
    long int k = ftell(stream);
    if(k<0){
        return -1;
    }
    char x[size];
    while(k>0 && fseek(stream, k-(long int)size, SEEK_SET)==0 &&
          fread(&x, size, 1, stream)==1 && cmp(x, ptr) != 0){
        k = k - (long int)size;
    }
    if(k==0 || ferror(stream)){
        return -1;
    }
    return k - (long int)size;
}
```

4 occ a

a	b	a	a	b
0	1	2	3	

ième(tab, a, r)   k = 0 ;   char x, nb1

a	b	a	b	c	b	a	c	b

```c
while (k<n && j <= nb1){
    if (t[k] == x){
        ++j; } ++k; }
if (j == nb1){
    return k;
} else {
    return NEANT;
}
```

j = 4

j = 6 != 4 ⊆ r

k = 4

4 — k;

l_n = = 0

j	k
2	2
3	3
3	4
3	5
4	6
5	7
5	8
5	9

Chap 06) Developements

Découpage + compilé séparée : (Donne lieu a des fichiers .h et .c.

.h = déclaration.
.c = définitions.

```
void str_all_stars (char *s) {

    char *ptr = s;
```

Chapitre 7 : Analyse des Algo.

notion 1 : Complexité.

- Temporelle : temps de calcul
- Spatiale : Espace mémoire requis.

Complexité pratique et théorique → ordre de ses coûts

mesure des complexités temporelles et spatiales

Problème des occurences répétées :

rapi si une lettre est répétée au moins 2 fois dans un mot.

Chapitre 8 : Algo numérique.

Multiplication Egyptienne :

234×37

- 1	234	$32+4+1$
2	468	
- 4	936	$7488+936+234$
8	1872	$=$
16	3744	$8658.$
- 32	7488	

x^{43}

1	x
2	x^2
4	x^4
8	x^8
16	x^{16}
32	x^{32}
	x^{43}

Horner.

$\sum_{k=0}^{n} a_k x^k$

Anneau commutatif, tout polynôme de degré n.

$$P = \sum_{k=0}^{n} a_k x^k.$$

$$((a) \cdot P)(x) = a + x\,P(x)$$

$$(P \cdot (a))(x) = P(x) + a x^{|P|}$$

Développement en base :

$$\lfloor \log_b n \rfloor = 0$$

$$0 \leq \lfloor \log_b n \rfloor \leq 1$$

$$1 \leq n \leq b-1$$

$$((a) \cdot P)(x) = a + xP(x)$$

$$P = (a_0, a_1, \dots a_n).$$

$$p(x) = a_0 + a_1 x^1 + a_2 x^2.$$

$$(\langle \gamma \rangle \cdot p)(x) =$$

$$(\langle \gamma \rangle \cdot (a_0, a_1, a_2 \dots a_n))(x).$$

$$(\gamma, a_0, a_1 \dots a_n)(x)$$

$$\gamma x^0 + a_0 x^1, a_1 x^2, a_2 x^3 \dots a_n x^{n+1})$$

$$\gamma + x\, p(x).$$

$$(p \cdot \langle \gamma \rangle)(x).$$

$$p = (a_0 \dots a_n)$$

$$((a_0 \dots a_n) \cdot \gamma)\, x$$

$$a_0 x^0 + \dots a_n x^n + \gamma x^{n+1}$$

$$\underline{a_0 + a_1 x^1 + a_2 x^2 \dots a_n x^n + \gamma x^{n+1}}$$

$$p(x) + \alpha x^{n+1} = p(x) + \gamma x^{\deg p + 1}.$$

$$x^{Tm+n} = x^{Tm} \cdot x^{Tn} = x^{Tn} x^{Tm}.$$

$$x^{Tmn} = (x^{Tm})^{Tn} = (x^{Tn})^{Tm}.$$

$$783 \times 43.$$

$$= 1 \quad 783$$

$$= 2 \quad 1566$$

$$4 \quad 3132 \qquad 33649.$$

$$= 8 \quad 6264$$

$$16 \quad 12528$$

$$= 32 \quad 25056$$

x^{123}

x^{123}

1	x
2	x^2
− 4	x^4
8	x^8
− 16	x^{11}
− 32	x^{32}
− 64	x^{64}
128	

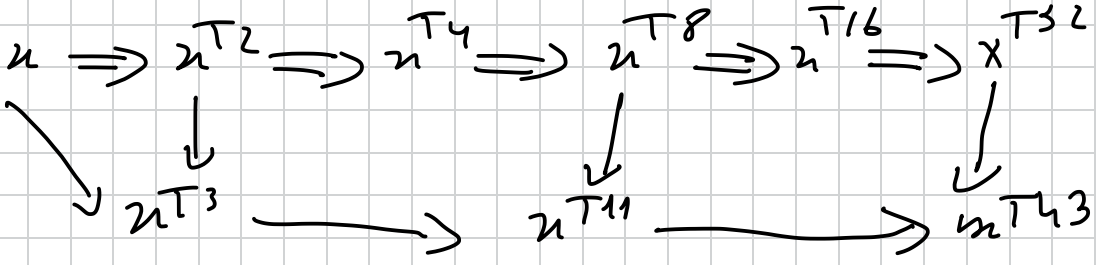
$x^{64} \times x^{32} \times x^8 \times x^{16} \times x^2 \times x = x^{123}$

$96+4=100$

$x \Rightarrow x^2 \Rightarrow x^4 \Rightarrow x^8 \Rightarrow x^{16}$

$x \quad x^{T2}$

$xT_3 = xTx^{T2}x^{T3}$



$d \leftarrow xTx$	$d = x^2$
$\Pi = xTd$	$\Lambda = x^3$ M $d = x^2$
$d \leftarrow dTd$	$d = x^4$ Λ $\Lambda = x^3$
$d \leftarrow dTd$	$d = x^8$ Λ $\Lambda = x^3$
$\Lambda \leftarrow dTM$	$d = x^8$ Λ $\Lambda = x^{11}$
$d \leftarrow dTd$	$d = x^{16}$ Λ $\Lambda = x^{11}$
$d \leftarrow dTd$	$d = x^{32}$ Λ $\Lambda = x^{11}$
$\Lambda \leftarrow \Gamma Td$	$d = x^{32}$ $\Lambda = x^{43}$

Si k impair

$r \leftarrow r \top d$

si $k \neq 1$ alors

$d \leftarrow d \top d$

$k \leftarrow \lfloor k/2 \rfloor$

$k = 123$; $r \leftarrow 1$; $d = x$

$r \leftarrow r \top d$) x

$d \leftarrow d \top d$) $d = x^2$ $x = x$

$r \leftarrow r \top d$) $d = //$ $x = x^3$

$d \leftarrow d \top d$

$d \leftarrow d \top d$)

$r \leftarrow r \top d$)

$d \leftarrow d \top d$

$d \leftarrow d \top d$)

$r \leftarrow r \top d$)

pour tout polyn $P = a_0 + a_1 x + \cdots a_n x^n$

$$\sum_{k=0}^{n} a_k x^k.$$

$$a_0 + x\left(a_1 + x^2\left(a_2 + x^3\left(\cdots a_{n-1} + x^{n-1} \times a_n \right)\right)\right)$$

A K C

C B A

$$X^{n \top M} = X^{\top n} \top X^{\top m} = X^{\top m} \top X^{\top m}$$

$$\left(X^{\top m n}\right) = \left(X^{\top n}\right)^{\top m} = \left(X^{\top m}\right)^{\top n}.$$

a	H	O	P	R	S	autre
d[a]	7	1	3	5	4	8

H	_	R	S	P	O	O	∀alphabet
7	6	5	4	3	2	1	0

K
S
P
O

{H}, {O, P, S, R}, {R}, {S}, {P}, {O}.

{O}, {∀alphabet}.

a	A	F	I	M	N	O	K	T	autre
d[a]	5	9	3	6	1	2	7	4	12

RΠAIΟN KMAIΟN

F R Π A N N O

P _ _ F IO K M A I I O N ∀a 12

11 10 9 8 7 6 5 4 3 2 1 0

A	C	G	T
1	2	J	J

T G	C, A	C	A	∀alphabet
4	3	2	1	0