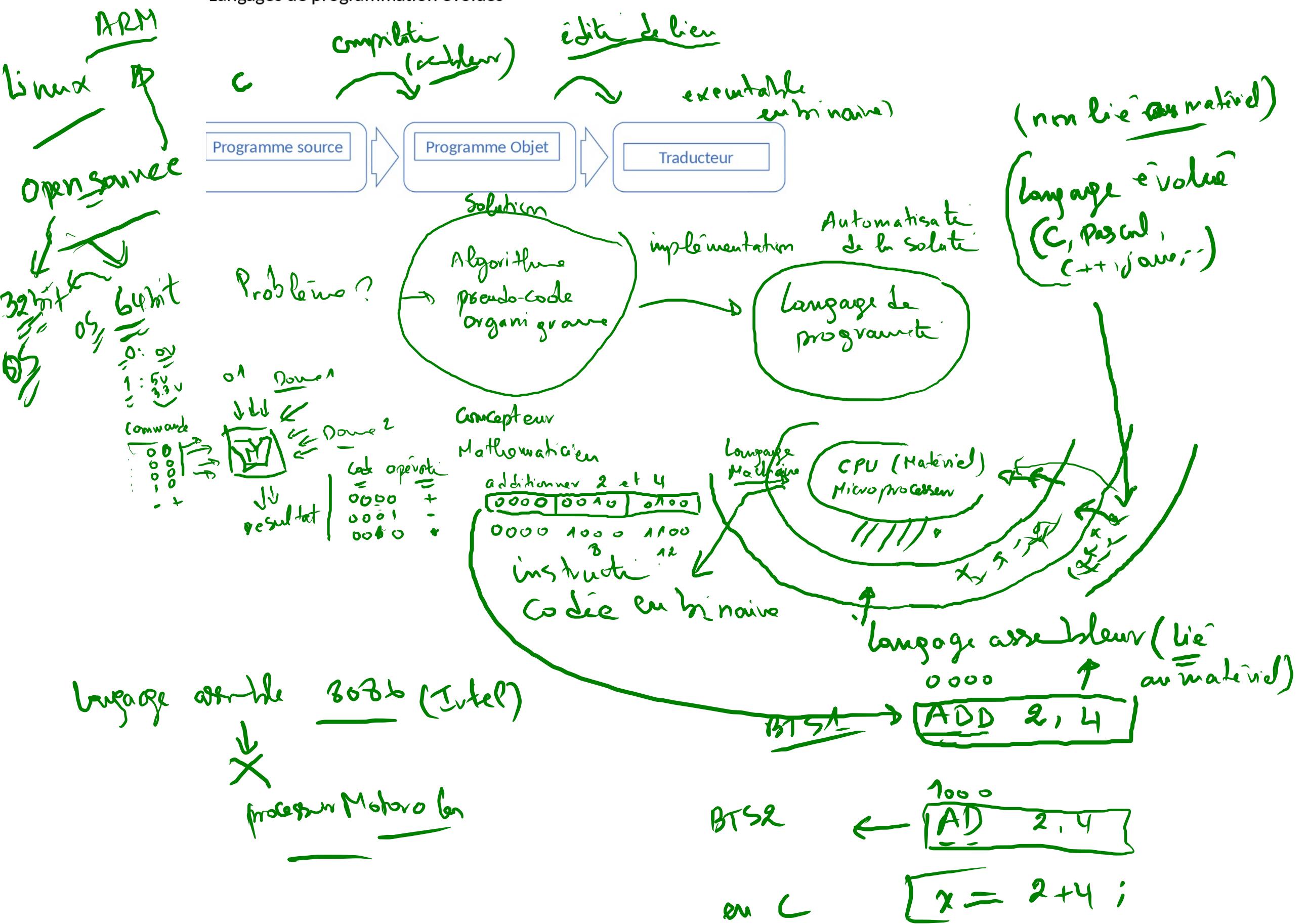
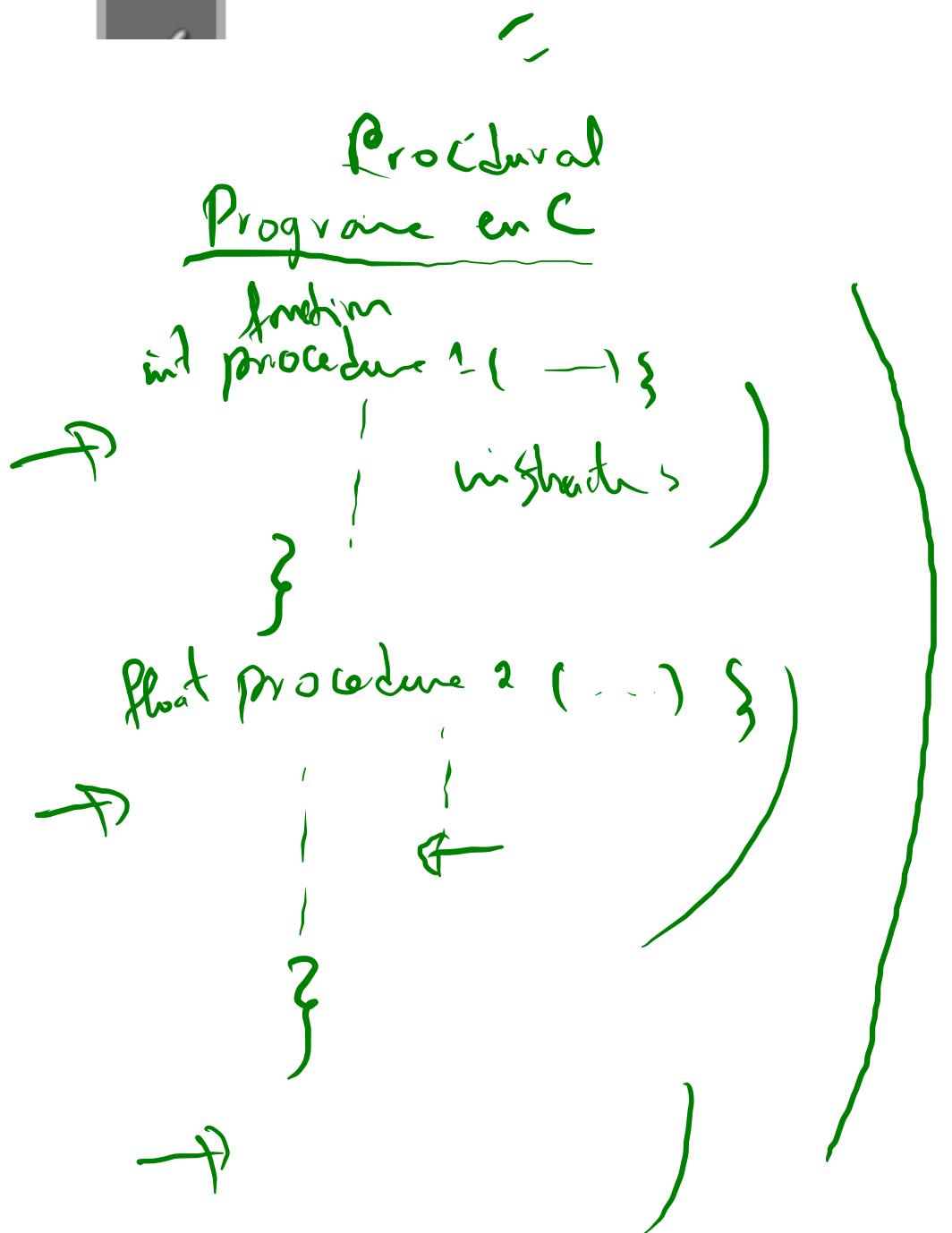


- Langage machine : opération sur les bits
 - Langage assembleur (symbolique) : opération sur les registres
 - Langages de programmation évolués



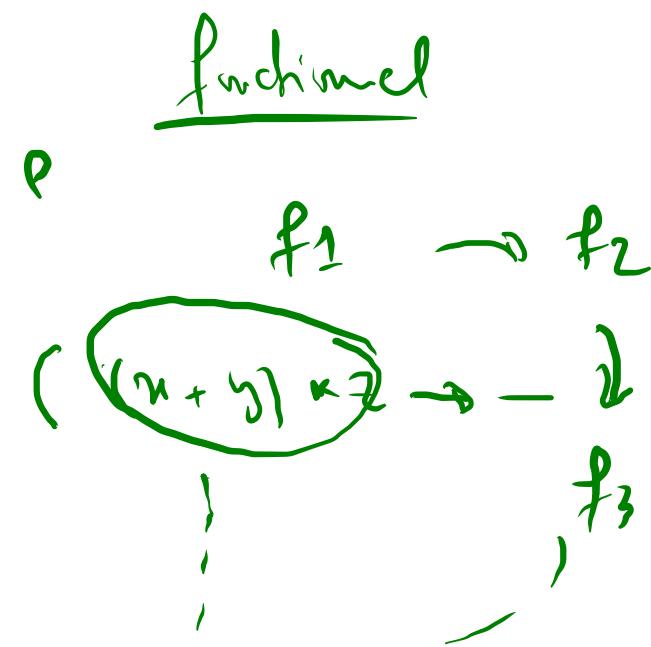
• Selon la structure interne

- procéduraux (C, C++, Pascal, etc.) : Le programme est une suite de procédures (instructions)
- déclaratifs - logiques (Prolog) : Le programme est une suite de propositions logiques
- fonctionnels (Lisp) : Le programme est une suite de fonctions



Déclaratif

- $a = (t) \downarrow ? \quad \leftarrow : P$
- $b : if(a) \quad \leftarrow$
-
-



Structure d'un programme en C

Entête :

Préprocesseurs,
Prototypes,
déclarations globales ...

main() {

Corps du programme

}

Windows



* éditeur

* compilateur



compilateur exécuter

CodeBlocks

Compilateur

(linux)

gcc exemple.c

exemple.o

éditeur de texte (Notepad / Notepad++, ...)

C++

exemple.c

exterieur

(exemple.cpp)

suite

scanf(" %d", &n);
printf(" le factorial de %d
est %d ", n, fact(n));

#include <stdio.h>
#define N 15

int factorial(int n){

}

void main(){

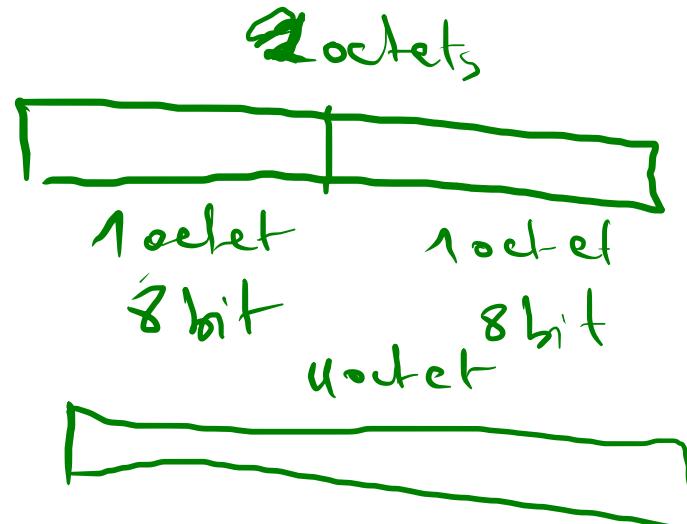
int n, p;

printf(" entre n: ");

les entiers

Nom de type	Autre Nom	Intervalle de valeur
int	signed, unsigned int	-32 768 à 32 767 $-\infty \rightarrow +\infty$
short	Short int, signed short signed short_int	-32 768 à 32 767 Δ
long	long_int, signed long, signed long_int	-2 147 493 648 à 2 147 483 647 $-\infty = +\infty$
unsigned	unsigned int	0 à 65 535
unsigned short	unsigned short_int	0 à 65 535
unsigned long	unsigned long_int	0 à 4 294 967 295

64 bit, (8 octets)
128 bits (16 octets)



int n;
n = 32767;

n = n+1;

$\Rightarrow \underline{32768X}$

negative Δ

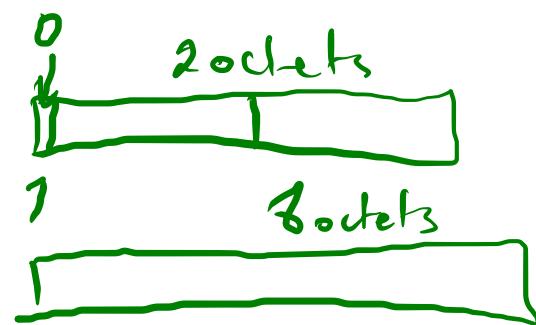
$\frac{+}{-} \in \mathbb{N}$

trajetoire

$n \leftarrow$
start



int n;
long ni;
long T[1000];
int B[1000];



$8 \times 1000 \text{ octets}$
 2×1000

$$\begin{aligned}2^{12} &= 2^6 \cdot 2^{10} \\&= 2^6 \text{ K} \\&= 64 \text{ K}\end{aligned}$$

Les nombres réels

float	-	-3.4E38 à +3.4E38
double	-	-1.7E308 à +1.7E308
long double	-	-1.7E308 à +1.7E308 DX

$$\begin{aligned} & \left[-3.4E38 \right] = -3.4 \times 10^{38} \\ & \text{float } x; \quad x \\ & \left. \begin{array}{l} +3.4E38 \\ +3.4 \times 10^{38} \end{array} \right\} \\ & \left[. , , \right] \\ & \underline{\underline{E_{rv}}} \end{aligned}$$

Les caractères

char	signed char	-128 à 127
unsigned char	-	0 à 255

char c ;

c = 100;

'c' = 'A';

Code ASCII de A est stocké dans c

char T[10];

'HAMID'

5 caractères

réserve de 10 caractères contiguous

T | H | A | M | I | D |

octet octet

10 octets

T[0] = 'H';

T[1] = 'A';

T[2] = 'M';

String s = "Hamid";



Nouveau type en C (#include <string.h>)

Exercices Algorithmiques

Exercice 0

Ecrire un programme qui échange la valeur de deux variables. Exemple, si $a = 2$ et $b = 5$, le programme donnera $a = 5$ et $b = 2$.

Début
a=2
b=5
 $t \leftarrow a$
 $a \leftarrow b$
 $b \leftarrow t$
Fin

ou bien Début
a=2
b=5
 $a \leftarrow a+b$
 $b \leftarrow a-b$
 $a \leftarrow a-b$
Fin

C est sensiblement la casse

$\text{maj} \neq \frac{\text{min}}{\text{c}}$

```
#include <stdio.h>
void main(){
    int a,b,t;
    a=8;
    b=5;
    printf("a=%d et b=%d avant l'echange.\n",a,b);
    //echange
    t=a;a=b;b=t;
    printf("a=%d et b=%d apres l'echange.\n",a,b);
    //printf(" a=2 et b=5 avant l'echange"); // A
    - t=a; a=b; b=t;
    printf(" a=%d et b=%d apres l'echange",a,b);
    scanf("%d", &n);
```

Exercice 1

Ecrire un programme qui demande un nombre à l'utilisateur, puis qui calcule et affiche le carré de ce nombre.

Variables nb, carr en Entier
Début
Ecrire "Entrez un nombre :"
Lire nb
 $carr \leftarrow nb * nb$
Ecrire "Son carré est : ", carr
Fin

En fait, on pourrait tout aussi bien économiser la variable carr en remplaçant les deux dernières lignes par :
Ecrire "Son carré est : ", nb*nb
C'est une question de style ; dans un cas, on privilégie la lisibilité de l'algorithme,

Exercices Algorithmiques

Exercice 0

Ecrire un programme qui échange la valeur de deux variables. Exemple, si $a = 2$ et $b = 5$, le programme donnera $a = 5$ et $b = 2$.

Début

~~a=2~~

~~b=5~~

$t \leftarrow a$

$a \leftarrow b$

$b \leftarrow t$

Fin

ou bien Début

$a=2$

$b=5$

$a \leftarrow a+b$

$b \leftarrow a-b$

$a \leftarrow a-b$

Fin

Exercice 1

Ecrire un programme qui demande un nombre à l'utilisateur, puis qui calcule et affiche le carré de ce nombre.

Variables nb, carr en Entier

Début

Ecrire "Entrez un nombre : "

Lire nb

$carr \leftarrow nb * nb$

Ecrire "Son carré est : ", carr

Fin

nb ??
carr

```
#include <stdio.h>
void main() {
    int nb, carre;
    printf("Entrez un nombre:");
    scanf("%d", &nb);
    carre = nb * nb;
    printf("Son carré est : %d", carre);}
```

En fait, on pourrait tout aussi bien économiser la variable carr en remplaçant les deux dernières lignes par :

Ecrire "Son carré est : ", nb*nb

C'est une question de style ; dans un cas, on privilégie la lisibilité de l'algorithme,

```
#include <stdio.h>
void main(){
    int nb, carre;
    printf("Entrer une valeur:");
    scanf("%d", &nb); // algo: lire nb;
    carre=nb*nb; // algo caree <- nb*nb ;
    printf("le carre de la valeur saisie est:%d\n", carre);}
```

dans l'autre, on privilégié l'économie d'une variable.

Exercice 2

Ecrire un programme qui lit le prix HT d'un article, le nombre d'articles et le taux de TVA, et qui fournit le prix total TTC correspondant. Faire en sorte que des libellés apparaissent clairement.

Variables nb, pht, ttva
Début
Ecrire "Entrez le prix hors taxe :"
Lire pht
Ecrire "Entrez le nombre d'articles :"
Lire nb
Ecrire "Entrez le taux de TVA :"
Lire ttva
pttc ← nb * pht * (1 + ttva)
Ecrire "Le prix toutes taxes est : ", nb * pht * (1 + ttva)
Fin
Là aussi, on pourrait squer :
Ecrire "Le prix toutes taxes est : ", nb * pht * (1 + ttva)
C'est plus rapide, plus léger en mémoire, mais un peu plus difficile à relire (et à écrire !)

```
#include <stdio.h>
void main(){
    int nb;
    float pht, tva, ptte;
    printf("Prix hors taxe:");
    scanf("%f",&pht);
    printf("Le nombre d'article:");
    scanf("%d",&nb);
    printf("TVA:");
    scanf("%f",&tva);
    ptte=nb*pht*(1+tva);
    printf("le prix TTC est:%f",ptte);
```

Exercice 3

Ecrire un algorithme qui demande deux nombres à l'utilisateur et l'informe ensuite si leur produit est négatif ou positif (on laisse de coté le cas où le produit est nul). Attention toutefois : on ne doit pas calculer le produit des deux nombres.

Variables m, n en Entier
Début
Ecrire "Entrez deux nombres :"
Lire m, n
Si (m > 0 ET n > 0) OU (m < 0 ET n < 0) Alors
Ecrire "Leur produit est positif"
Sinon
Ecrire "Leur produit est négatif"

Finsi
Fin

Exercice 4

Ecrire un algorithme qui demande l'âge d'un enfant à l'utilisateur. Ensuite, il l'informe de sa catégorie :

- * "Poussin" de 6 à 7 ans
- "Pupille" de 8 à 9 ans
- "Minime" de 10 à 11 ans
- "Cadet" après 12 ans

Peut-on concevoir plusieurs algorithmes équivalents menant à ce résultat ?

```
Variable age en Entier
Début
Ecrire "Entrez l'âge de l'enfant : "
Lire age
Si age >= 12 Alors
Ecrire "Catégorie Cadet"
SinonSi age >= 10 Alors
Ecrire "Catégorie Minime"
SinonSi age >= 8 Alors
Ecrire "Catégorie Pupille"
SinonSi age >= 6 Alors
Ecrire "Catégorie Poussin"
Finsi
Fin
```

On peut évidemment écrire cet algorithme de différentes façons, ne serait-ce qu'en commençant par la catégorie la plus jeune.

Exercice 5

Ecrire un algorithme qui demande un nombre compris entre 10 et 20, jusqu'à ce que la réponse convienne. En cas de réponse supérieure à 20, on fera apparaître un message : *Plus petit !*, et inversement, *Plus grand !* si le nombre est inférieur à 10.

```

    Variable N en Entier
Debut
N ← 0
Ecrire "Entrez un nombre entre 10 et 20"
Lire N
TantQue N < 10 ou N > 20
Si N < 10 Alors
Ecrire "Plus grand !"
Sinon Si N > 20 Alors
Ecrire "Plus petit !"
FinSi
FinTantQue
Fin

```

Exercice 6

Ecrire un algorithme qui demande un nombre de départ, et qui calcule la somme des entiers jusqu'à ce nombre. Par exemple, si l'on entre 5, le programme doit calculer : $1 + 2 + 3 + 4 + 5 = 15$ NB : on souhaite afficher uniquement le résultat, pas la décomposition du calcul.

```

Variables N, i, Som en Entier
Debut
Ecrire "Entrez un nombre : "
Lire N
Som ← 0
Pour i ← 1 à N
Som ← Som + i
i Suivant
Ecrire "La somme est : ", Som
Fin

```

Exercice 7

Ecrire un algorithme qui demande un nombre de départ, et qui calcule sa factorielle.

NB : la factorielle de 8, notée $8 !$, vaut $1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 7 \times 8$

Variables N, i, F en Entier
Debut
Ecrire "Entrez un nombre : "
Lire N
 $F \leftarrow 1$
Pour $i \leftarrow 2$ à N
 $F \leftarrow F * i$
i Suivant
Ecrire "La factorielle est : ", F
Fin

Exercice 8

Que produit l'algorithme suivant ?
Tableau Nb(5) en Entier
Variable i en Entier
Début
Pour $i \leftarrow 0$ à 5
 $Nb(i) \leftarrow i * i$
i suivant
Pour $i \leftarrow 0$ à 5
Ecrire Nb(i)
i suivant
Fin
Peut-on simplifier cet algorithme avec le même résultat ?

Cet algorithme remplit un tableau avec six valeurs : 0, 1, 4, 9, 16, 25.
Il les écrit ensuite à l'écran. Simplification :
Tableau Nb(5) en Numérique
Variable i en Numérique
Début
Pour $i \leftarrow 0$ à 5
 $Nb(i) \leftarrow i * i$
Ecrire Nb(i)
i Suivant
Fin

Exercice 9

écrivez un algorithme permettant, à l'utilisateur de saisir les notes d'une classe. Le programme, une fois la saisie terminée, renvoie le nombre de ces notes supérieures à la moyenne de la classe.

Variables Nb, i, Som, Moy, Nbsup en Numérique
Tableau T() en Numérique
Debut
Ecrire "Entrez le nombre de notes à saisir : "
Lire Nb
Pour i \leftarrow 1 à Nb
Ecrire "Entrez le nombre numero", i
Lire T(i)
i Suivant
Som \leftarrow 0
Pour i \leftarrow 1 à Nb
Som \leftarrow Som + T(i)
i Suivant
Moy \leftarrow Som / Nb
NbSup \leftarrow 0
Pour i \leftarrow 1 à Nb
Si T(i) > Moy Alors
NbSup \leftarrow NbSup + 1
FinSi
i Suivant
Ecrire NbSup, " élèves dépassent la moyenne de la classe"
Fin