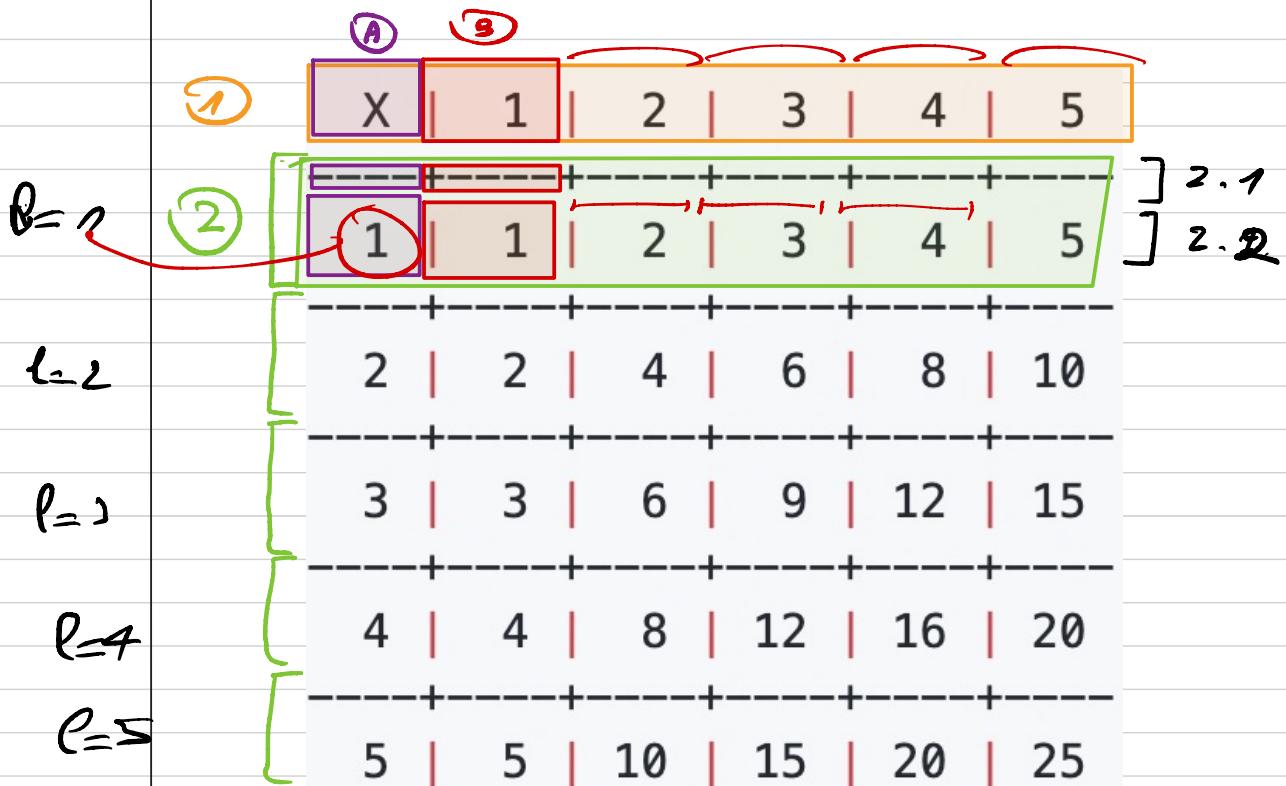


Labs 10 - Analyse

30. XI. 2022

$N=5$



Affiché par ligne et envoi au colonne.

② = $f(e)$ fait Nx

① fait $1x$

l'ordre de la colonne = $f(N^2)$

$$\log_{10}(N^2)$$

$$N=5$$

$$N=10$$

$$N^2=25$$

$$N^2=100$$

$$\log_{10}(N^2) = 1.39$$

$$= 2.00$$

+2 → l'ordre de N^L .

$$N = 1 + (\text{int}) \log_{10}(N \times N)$$

for ($l=0$; $l < N$; $l++$) {

 for ($c=0$; $c < N$; $c++$) {

 if ($c == 0$) { ←

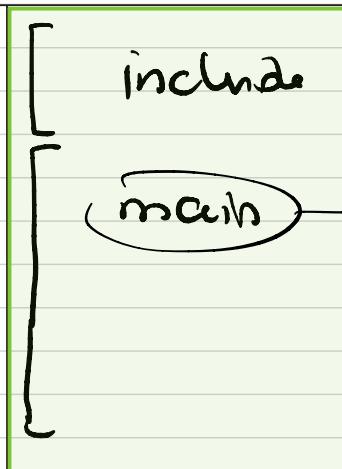
 }

 }

} ←

les fonctions

30. XI. 2020



fonction principale de
votre programme

$x = \cos(\text{angle});$

:

$y = \cos(\text{angle2});$

:

fonction

découper le
traitement en
fonctions

pointf ssant

angle

double
radian

paramètre
d'entrée

cos

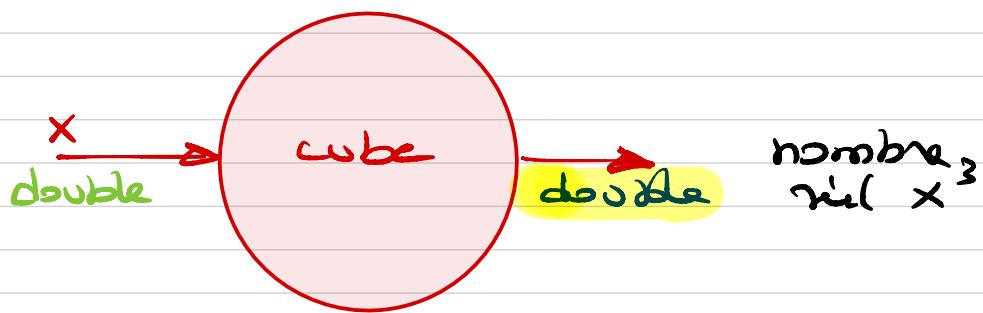
double
[-1...+1]

cosinus de
l'angle

valeur de sortie.

Fonction pour calculer le cube d'un nombre. "cube"

nombre réel x



① l'interface ou le prototype : ↗ ligne de code qui définit comment utiliser la fonction

`double cube (double x);`

// on présente la fonction.

② l'implémentation de la fonction.

`double cube (double x) {`

`double result = 0.;`

`result = x * x * x ;`

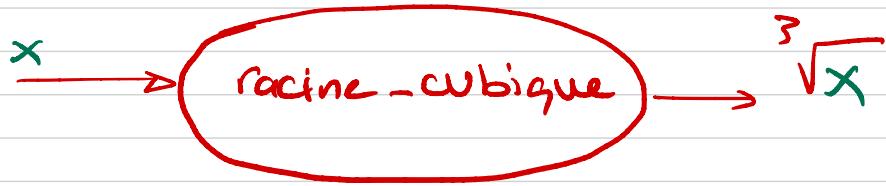
`} return result;`

③ utilisation de la fonction

`double z=3.;`
`double r=0.;`

`r = cube (z);`

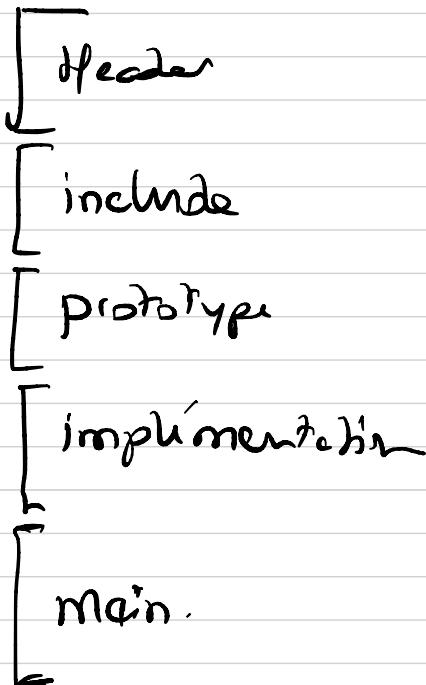
TD 2020/130



- ① Écrire le prototype
- ② Écrire l'implémentation de la fonction
- ③ Écrire le main et de test pour

$$x = 27$$

TD 2020/130.c



13' 58

Passage par valeurs

```

int main(int argc, char const *argv[])
{
    double z = 27.;
    double r = 0.;

    r = racine_cubique(z);

    printf("z=%lf    r=%lf\n", z, r);
    result = 3.0;
    return 0;
}

```

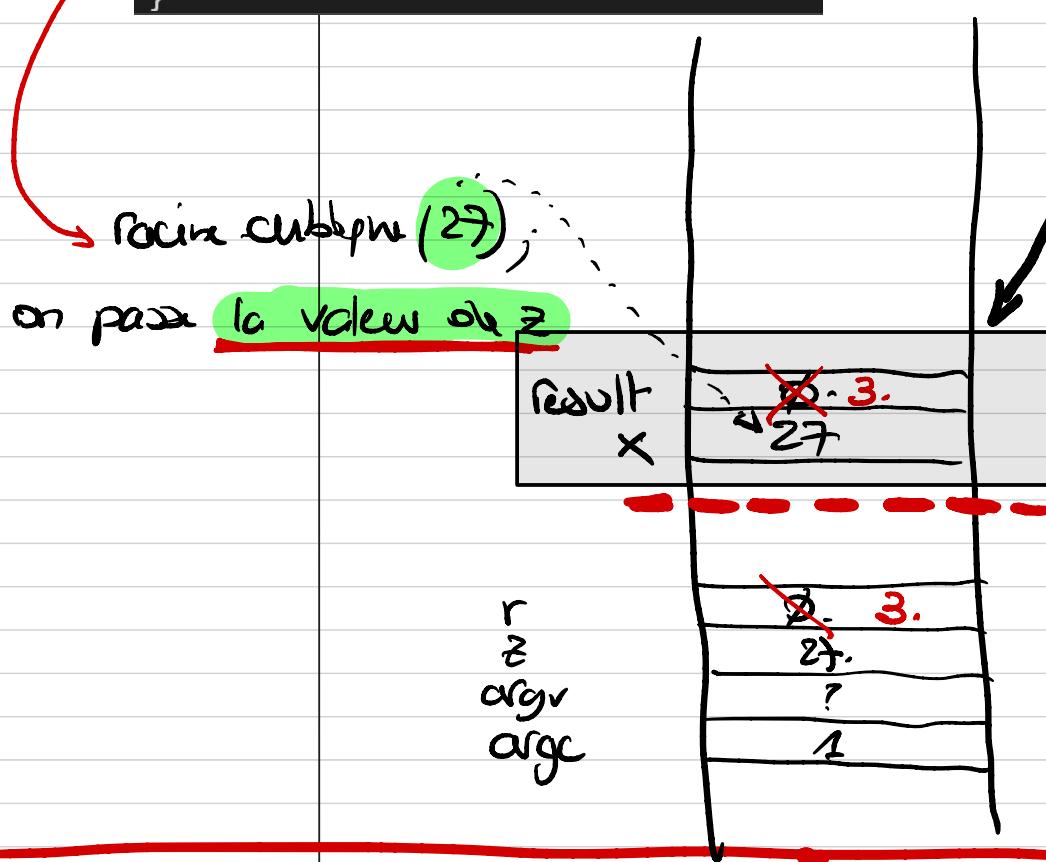
```

// implementation
double racine_cubique(double x) {
    double result = 0.;

    result = pow(x, 1./3.);

    return result; 3.0
}

```



main:
`argc`
`argv`
`z`
`r`

`argc`, `argv`, `z` et `r` sont des variables locales à `main`

racine_cubique:
`x`
`result`

`x` et `result` sont des variables locales à `racine_cubique`

[Améliorer la fonction racine-cube pour une utilisation $-\infty \rightarrow +\infty$ pour x ,

\mapsto^{h27}

Gérer la formule x .

$$\text{si } x \geq 0 \rightarrow \sqrt[3]{x}$$

$$\text{si } x < 0 \rightarrow -\sqrt[3]{-x}$$

$$\begin{array}{c} \text{signe } x \\ \text{---} \\ \sqrt[3]{|x|} \end{array}$$

$$\text{result} = \sqrt[3]{|x|}$$

si $x \geq 0$

return result

sinon

return - result

$$\frac{x}{|x|} \cdot \sqrt[3]{|x|}$$

$$\begin{array}{l} \sqrt[3]{x} \\ 2x \\ 1x \\ 1x \end{array}$$

diviser

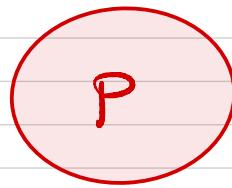
multiplié.

$$\begin{array}{l} \sqrt[3]{x} \\ 1x \\ 1x \\ 1x \end{array}$$

opp.

Fonction / paramètre

Pas de param
en entrée



Pas de résultat
en sortie.

procédure

void p(void);

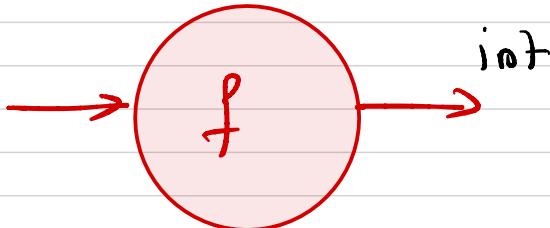
void p(void) {

==

return ;

f

double k



fonction à 1 param. en entrée

int f(double k);

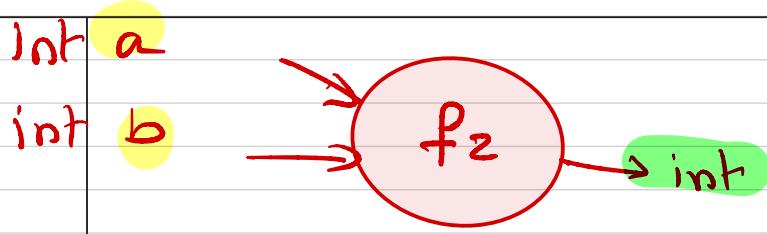
int f(double k) {

int result = 0;

==

return result;

f



fonction avec 2 param. en entrée
et 1 entier en retour

int f2(int a , int b) ;

int r=φ;

r = f2(42 , 3);

a b

