

Info2 - Complément pour le laboratoire

11. DS.2a.2)

③ Accès par champs

r = scanf (" %d %d ", &i, &j);

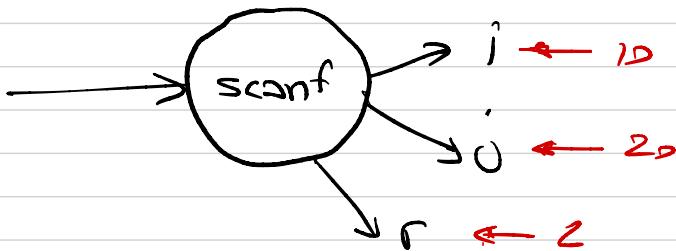
Clavier

stdin

standard

input

10 20 ↴



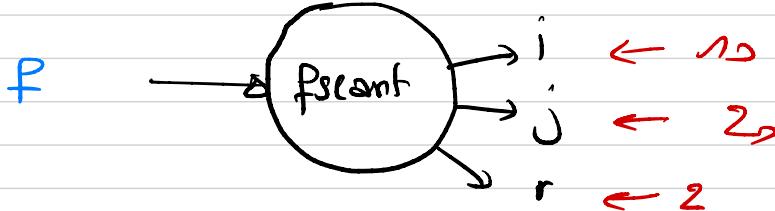
(!) toujours vérifier le retour du scanf.

f = fopen ("data.txt", "r");

r = fscanf (f, "%d %d", &i, &j);

data.txt

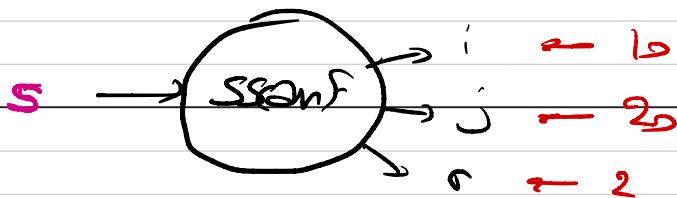
10 20



(!) toujours vérifier le retour du scanf.

char * s = "10 20";

r = sscanf (s, "%d %d", &i, &j);



(!) toujours vérifier le retour du scanf.

data.txt



```
int i = 42;  
int j = 37;  
int r = 1;
```

```
f = fopen("data.txt", "r");
```

Option 1

```
r = fscanf(f, "%d %d", &i, &j);
```

	Avant	Après
i	42	1234
j	37	37

```
#define N 1024
```

```
char s[N];
```

```
fgets(s, N, f); // s ← "1234"
```

```
r = sscanf(s, "%d %d", &i, &j);
```

```
i ← 1  
j ← 1234  
j ← 37
```

data.txt

ABCD
1234
XYZT

ABCD\n1234\nXYZT ~~E~~

FILE *f = NULL;

f = fopen("data.txt", "r");

// Test if f != NULL

char s[N];

fgets(s, N, f); s ← "ABCD\n"

fgets(s, N, f); s ← "1234\n"

fgets(s, N, f); s ← "XYZT"

chain	int	
rouge	1	
bleu	2	
vert	3	

int coupl = 2;

typedef enum {

ROUGE = 1,
BLEU = 2,
VERT = 3 } int;

} COULEUR;

COULEUR c = ROUGE;

if (c == VERT) { ... }

printf ("c=%d", c); c = 1

typedef enum {

NO_ERROR = 0,
BAD_FORMAT = 1,

;

} ErrorCode;

Analyse.

18266 lines (18265 sloc) | 571 KB

1	MeteoSchweiz/MeteoSuisse/MeteoSvizzera/MeteoSwiss
2	
3	stn time tre200d0
4	LSN 19310101 5.3
5	LSN 19310102 5.5
6	LSN 19310103 8.4
7	LSN 19310104 5.1
8	LSN 19310105 1.4
9	LSN 19310106 -1.2
10	LSN 19310107 -2.8
11	LSN 19310108 -2.9
12	LSN 19310109 -5.0
13	LSN 19310110 -5.9
14	LSN 19310111 -6.8
15	LSN 19310112 -5.1
16	LSN 19310113 -2.2

3 premières lignes

① "LSN
annee

19310101

5.3\n"
temp

② Comment faire

1) l'accumulation des temp pour un
m année

2) les transitions pour les années

③ Calcul valeur moyenne et affichage

char

$s_1[10] = "Hello";$

char

$s_2[10] = "";$

strcpy(s_2, s_1);

10 char

s_2

10 char

s_1

10

10

10

10

10

10

1000

loop k=28

char $s_1[10] = "Hello";$

char s_2 = NULL;

strcpy(s_2, s_1);

s_2

280

1000

10 char

s_1

10

10

10

1000

k

800

(),

```
int f( void ) {
```

==

```
return [ ] int ;
```

}

root f(void) {

==

return;

}

./app ~

usage:

./app

— —

./app — —

" ./app "

main:

```
usage ( argc [ ] );
```

```
usage ( char * appNm ) {
```

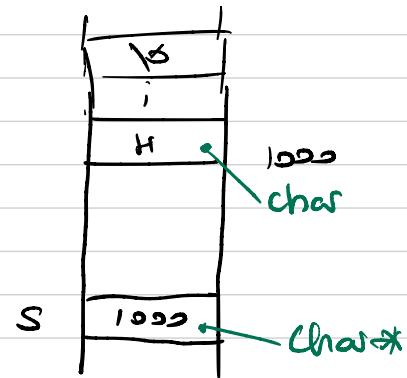
```
printf ( " %s ... ", appNm );
```

chaînes et pointeurs, quel bonheur !

11. III. 2021

①

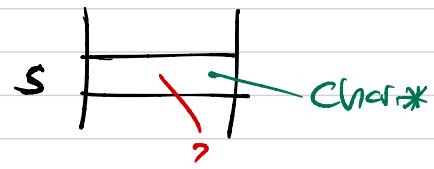
`char *s = "Hi";`



②

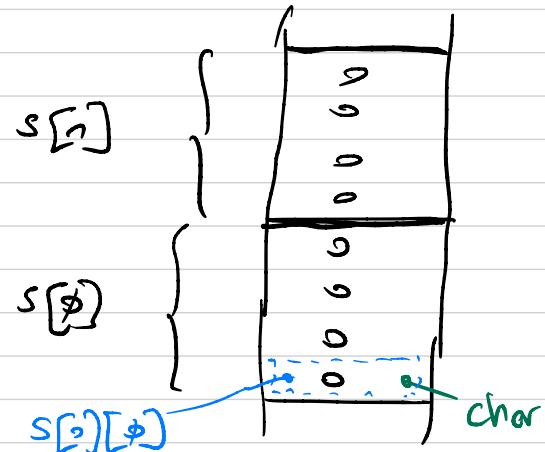
`char *s;`

(!) valeur indéfinie.
danger ...



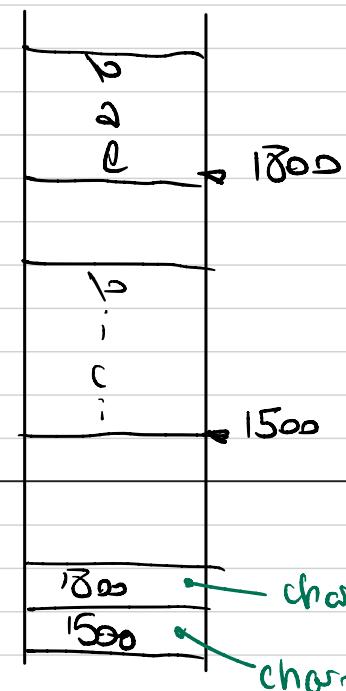
③

`char s[2][4] = {{'f', 'p'},`



④

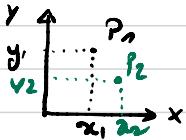
`char *s[2] = {"ici", "la"};`



Préparation cours du M. Ju. 2021

Préambule aux fichiers binaires

Les structures de données



doubs x_1, y_1, x_2, y_2 .

fonction pour afficher les coordonnées d'un point (2 param)

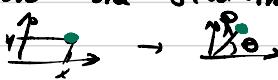
fonction pour calculer la distance entre 2 points (4 param)

Concept d'une structure

type / champs / nom

Refaire l'exemple avec une struct.

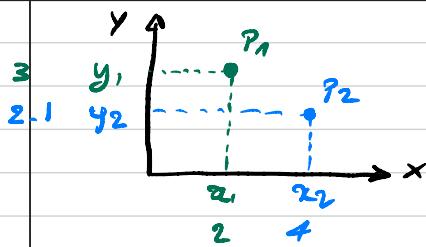
Pointeur sur une structure

Exemple  → 

[struct point 1D cotéien
struct point 2D point]

Sinfo2 M: - structures

11. 04. 2021



x₁, x₂, y₁ et y₂ sont des nœuds.

$$\Delta x = 2 \\ \Delta y = 0.9$$

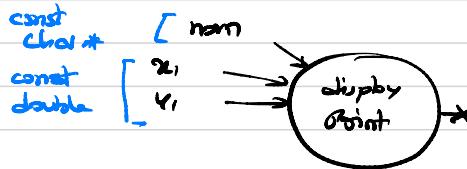
TD 2021/03/11:

- 1) créer les 4 variables x₁, y₁, x₂ & y₂
+ valeurs initiales
- 2) appeler une fonction "displayPoint"
qui affiche les coordonnées x et y
d'un point et son nom
 $P_1: (+2.0, +3.0)$
- 3) appeler une fonction qui calcule
et affiche la distance entre 2 points
"computeDistance"

Analyse → prototype
→ pseudo-code.

16' 55.

① display Point



`void displayPoint (const char * name, const double x, const double y);`

② compute Distance



`double computeDistance (const double x1, const double y1,
const double x2, const double y2);`

$$\text{Calcul} \rightarrow (\text{Pythagore}) \quad \text{distance} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Implémentation des 2 fonctions + test avec les valeurs en haut de la page.

17' 11

Point 2D

coordonnée x
coordonnée y
nom

double
double
char [...]

Structur en c → 1 nouveau type → choisir son nom
→ des données "cohérentes" entre elles
↳ "champs" → type et un nom.

Structur: sPoint2D

↳ x
↳ y
↳ name

← nom du nouveau type
type double
double
char [20]

```
typedef struct {  
    double x;  
    double y;  
    char name [20];  
} sPoint2D;
```

création d'un nouveau type

L'initialisation des valeurs
des champs s'effectue lors
de la création des variables.

Utilisation de la structur

sPoint2D p1 = { 2., 3., "P1" };
sPoint2D p2 = { 4., 2.1, "P2" };] création de 2 variables structurées

printf ("x = %f", p1.x);

p1.y

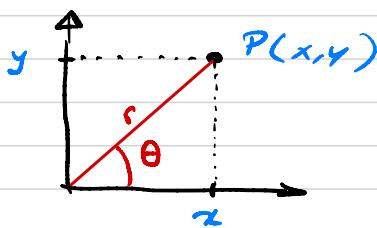
%s p1.name

TD 2021/03/11 b

1) créer la structure

2) adapter le TD 2021/03/11 aux le type structuré.

17/48



c2p: Cartesian to Polar
(x, y) \rightarrow (r, θ)

Créer une fonction telle que:



TD 20210311c

- 1) créer la fonction c2p
- 2) ajouter un type struct pour le modèle polarique (r, θ)
- 3) afficher le (r, θ) pour les points P_1 et P_2 du TD 20210311 b.

A terminer pour la session du 15. III. 2021 -

