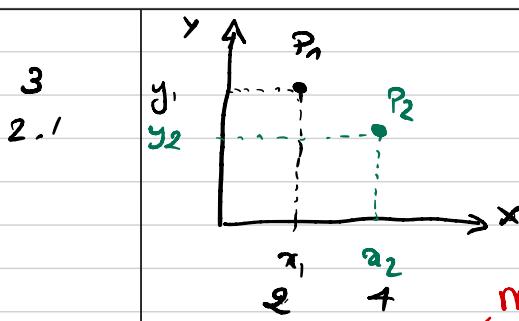


structures

IS. Th. 2021



x₁, y₁, x₂, y₂ → 4 variables "double"

displayPoint

TD20210315:

main

- 1) création des 4 variables + init
- 2) 1 fonction pour afficher le nom et les coordonnées d'un point

P₁ +2.000 +3.000

computeDistance

- 3) 1 fonction pour calculer et renvoyer la distance entre 2 points

→ Afficher P₁ P₂ et la distance entre les 2 points

Analyse

- 1) création du prototype
- 2) décider de l'implémentation

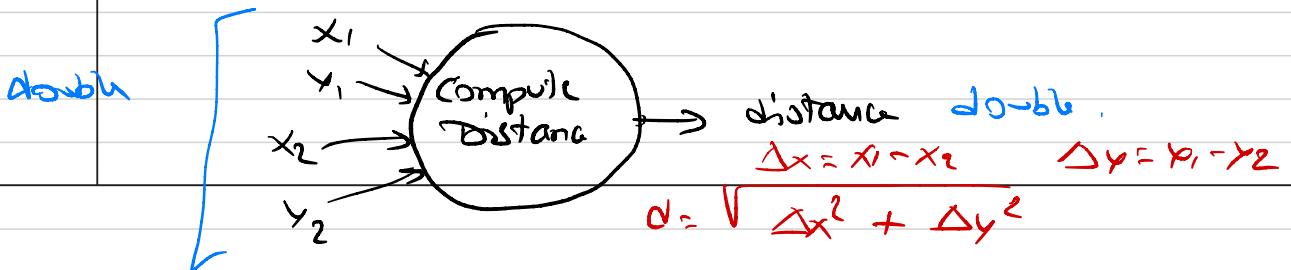
char *
nom du point
double x
double y

display
Point

3+0.3 Pf

void

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



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

(!) #include <math.h>

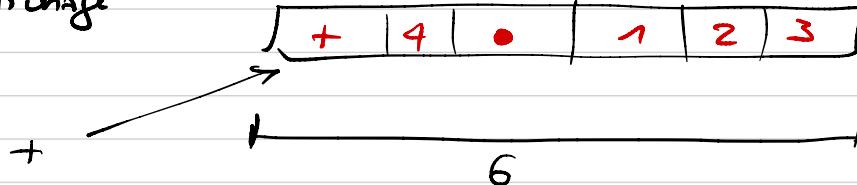
pour l'affichage du signe

Format % + 6.3 pp ...

largeur totale
de l'affichage

nbre de chiffres
après la virgule.

3



double $x = 1234567$

+1234567.000

Structure

Rassembler dans un nouveau type de donnée des informations cohérentes entre-elles

Point: $x, y, \text{ nom}$

Reptition tel: $\text{nom}, \text{ prénom}, \text{n}^{\circ} \text{ tel}$

Création:

- 1) nom du nouveau type
- 2) pour chaque information cohérente:
 - ↳ type de l'info
 - ↳ son nom

typedef struct {

double $x,$
double $y;$
char * name;

} sPoint2D;

un nouveau type

sPoint2D p1 = {2., 3., "P1"};

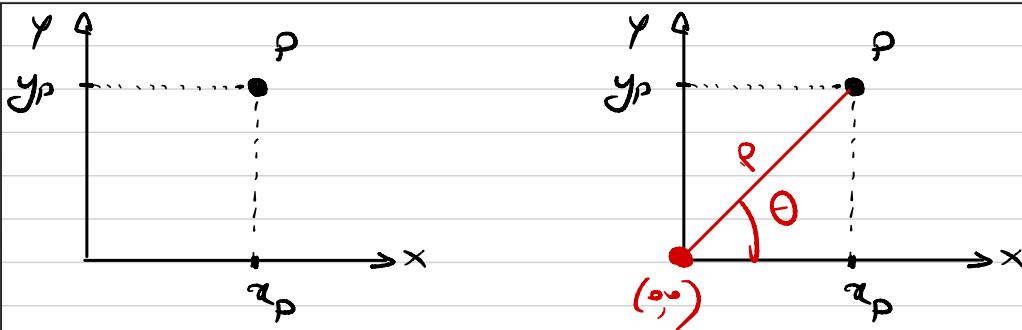
sPoint2D p2 = {4., 2.1, "P2"};

x y name

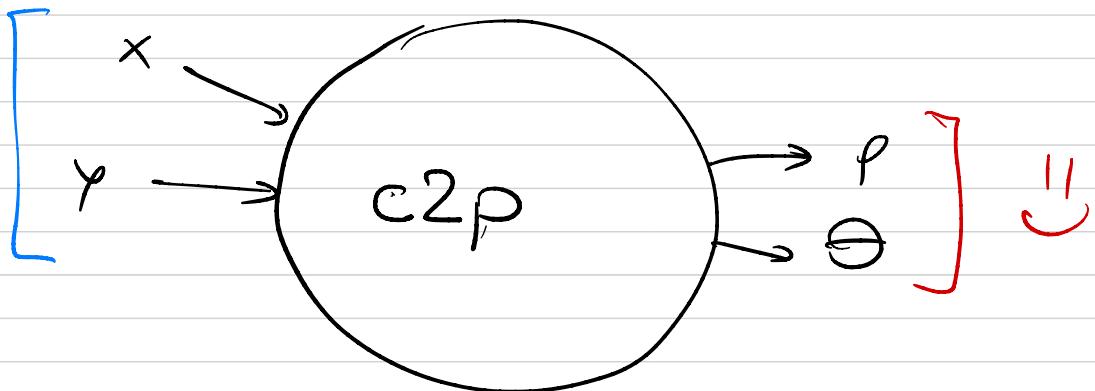
sPoint2D p3;

p3 = p1; // copie tous les champs.

printf ("% +6.3lf\n", p1.x); / +2.000
p3.x = 42; /



sPoint2D



- ① Comment passer du monde cartésien au monde polaire
- ② Types de données !!
- ③ main: afficher les coord. polaires de p1 et p2

Analyse

$$r = \sqrt{x_p^2 + y_p^2}$$

$$\theta = \arctan\left(\frac{y}{x}\right)$$

$$\left(-\frac{\pi}{2} \dots +\frac{\pi}{2}\right)$$

$$\theta = \text{atan2}(y, x)$$

$$(-\pi \dots +\pi)$$

void f (sPoint2D &P) {

 printf ("x= %f", P->x);

P "pointe" sur une structure.

labo38c

#include < ... >

#define

typedef struct ...

prototypes ...

=====

fonctions ...

Suivi du TD: → TD20210315b

- ① Création structure sPoint2D
- ② Adapter les 2 fonctions `displayPoint` et `computeDistance` pour utiliser le type structure
- ③ Modifier le main
- ④ Composer les réponses avec TD 20210315.