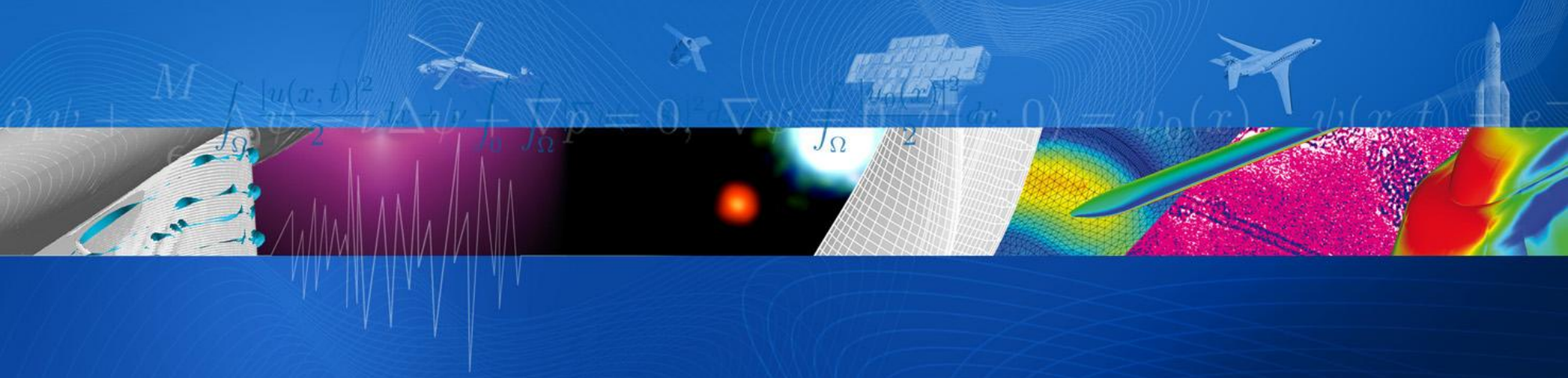


Introduction à la programmation C++

Les tableaux statiques

BOULCH Alexandre

ONERA
THE FRENCH AEROSPACE LAB

retour sur innovation

Plan de la séance

Les tableaux

Des tableaux pour . . .

- . . . éviter la multiplication des variables
- . . . structurer les données (e.g. coordonnées d'un vecteur)
- . . . parcourir rapidement un ensemble d'éléments

Utilisation

- Les tableaux ont un type
- Les tableaux ont une taille fixe (une constante)

Déclaration

```
type nom_tableau[taille];
```

Initialisation

C++

```
int tab[10];
for(int i=0; i<10; i++){
    tab[i] = 5;
}

double tab2[5] = {2,3.2,9.76,6,1000};

bool tab3[3];
tab3 = {true,true,false}; // ERREUR
```

Comparaison avec Python

Python

```python
tab1 = [0 for i in range(5)]
tab1[2] = 5

tab2 = ["test", 10, True]

t = 6
tab3 = [0 for i in range(t)]
tab3.append(100)
```

C++

```cpp
int tab1[5] ={0,0,0,0,0};
tab1[2] = 5

bool tab1 ={"test", 10, True}; // ERREUR

int t = 6
int tab3[t]; // ERREUR t non constant

const int t = 6;
int tab3[t]; // OK t constant
tab3.append(100) // ERREUR taille fixe
```

N.B.

Les tableaux en C++ sont plus proches des tableaux numpy que des listes python.

Taille constante

Pour créer un tableau il est impératif que la taille soit une constante : un nombre ou un `const int`.

C++

```
int tab1[5]; // OK

const int taille = 1000; // declaration d'une constante
double tab2[taille];
```

C++ Erreurs

```
double taille2 = 100;
float tab3[taille2]; // ERREUR la taille est un double

int taille3 = 200;
float tab3[taille3]; // ERREUR la taille n'est pas une constante
```

Utilisation

Si `n` est la taille du tableau, les indices vont de `0` à `n-1`.

C++

```
const int n=100;
char tab[n];
tab[0] = 'a'; // OK
tab[n] = 'f'; // ERREUR
```

Les tableaux utilisent de la mémoire, ne pas les utiliser si ils ne sont pas nécessaires.

C++

```
//calcul 2^99
int t[100];
t[0] = 1;
for(int i=1; i<100; i++){
    t[i] = t[i-1]*2;
}
cout << t[99] << endl;
```

C++

```
//calcul 2^99
//sans tableau
int r=1;
for(int i=1; i<100; i++){
    r *= 2;
}
cout << r << endl;
```

Plan de la séance

Tableaux et fonctions

On peut utiliser les tableaux dans les fonctions :

C++

```
void affiche(int t[5]){
    for(int i=0; i<5; i++){
        cout << t[i] << " ";
    }
    cout << endl;
}
```

C++

```
void affiche(int t[], int taille){
    for(int i=0; i<taille; i++){
        cout << t[i] << " ";
    }
    cout << endl;
}
```

La seconde solution est à préférer car elle réutilisable avec des tableaux de différentes tailles.

Tableaux et fonctions 2

Attention

- Un tableau est **toujours** passé par référence.
  On n'utilise pas de &.
- Une fonction ne peut pas retourner de tableau.

C++

```
const int taille = 10;
double tab[taille];
init(tab);
affiche(tab);
// 0 0 0 0 0 0 0 0 0 0
```

C++

```
void init(double t[], int taille){
    for(int i=0; i<taille; i++){
        t[i] = 0;
    }
}
```

Égalité de tableaux

On ne peut pas faire d'égalité entre les tableaux.

C++

```
bool t1[4]={1,2,3,4}, t2[4];
t2 = t1 ; // ERREUR : pas d'affectation avec le = pour les tableaux
```

Seule solution : itérer sur les éléments.

C++

```
bool t1[4]={1,2,3,4}, t2[4];
for(int a=0; a<4; a++){
    t2[a] = t1[a];
}
```

Plan de la séance

Les modules

- **Common**
  fonctions et classes basiques (Timer, Color...)
- **LinAlg**
  algèbre linéaire (inversion de matrices...)
- **Graphics**
  affichage (fenêtre 2D/3D, dessin...)
- **Images**
  classe Image et traitements

Programme simple

C++

```
#include<iostream>
#include <Imagine/Graphics.h>
using namespace std;
using namespace Imagine;

int main(){
    int xc=128, yc=128,t=0,r; // init variables

    openWindow(256,256); // Ouverture de la fenetre

    while (true) { // Boucle principale

        r = 10*cos(t/1000); // mise a jour du rayon

        fillCircle(xc,yc,r,RED); // Affichage du disque

        milliSleep(20); // Temporisation

        fillCircle(xc,yc,r,WHITE);// Effacement du disque

        t++; // incremente le temps
    }
    endGraphics();
    return 0;
}
```

Gestion des fenêtres

Il est possible d'ouvrir et de travailler avec plusieurs fenêtres graphiques.

C++

```
// premiere fenetre
openWindow(256,256);
fillCircle(128,128,50,RED);

// seconde fenetre
openWindow(256,256);
fillCircle(128,128,50,BLUE);

//
//impossible de revenir dessiner
//dans la premiere fenetre :
//elle n'a pas de nom
//
```

C++

```
// premiere fenetre
Window window1 = openWindow(256,256);
fillCircle(128,128,50,RED);

// seconde fenetre
Window window2 = openWindow(256,256);
fillCircle(128,128,50,BLUE);

setActiveWindow(window1);
fillCircle(128,128,50,GREEN);

setActiveWindow(window2);
fillCircle(128,128,50,BLACK);

// fermeture d'une fenetre
closeWindow(window1);
```

La documentation

Le site du cours → Installation Imagine++ → Instructions

Imagine++

Main Page | Related Pages | Modules | Namespaces | Classes | Files | Examples

Imagine++ Libraries - version 4.2.0

The most up to date version of this documentation should be on the website: http://imagine.enpc.fr/~monasse/Imagine++/

Introduction

Imagine++ is a set of libraries developed at the Imagine group (http://imagine.enpc.fr). Initially designed for students and beginners, and though it is still easy enough for them, Imagine++ is now used daily by Imagine researchers. It consists in different modules. Four of them are publicly available:

- The **Common Library**, providing basic types and utilities.
- The **LinAlg Library**, providing linear algebra types and algorithms
- The **Graphics Library**, providing convenient 2D and 3D displays
- The **Images Library**, providing image containers and algorithms

Some useful links:

- The Imagine++ home page is http://imagine.enpc.fr/~monasse/Imagine++
- A quick start guide
- This C++ course for beginners (in French...) makes extensive use of Imagine++, especially of the Graphics module: http://imagine.enpc.fr/~monasse/Info/
- For any question, feel free to contact monasse@imagine.enpc.fr

Installation

- **Windows installation**

La documentation

Le site du cours → Installation Imagine++ → Instructions

void	**Imagine::drawRect** (const IntPoint2 &p, int w, int h, const Color &col, int penWidth=1, bool xorMode=false) Rectangle (IntPoint2). More...
void	**Imagine::drawString** (int x, int y, const std::string &s, const AlphaColor &col, int fontSize=12, double alpha=0, bool italic=false, bool bold=false, bool underlined=false, bool xorMode=false) String. More...
void	**Imagine::drawString** (const IntPoint2 &p, const std::string &s, const AlphaColor &col, int fontSize=12, double alpha=0, bool italic=false, bool bold=false, bool underlined=false, bool xorMode=false) String (IntPoint2). More...
void	**Imagine::enableMouseTracking** (bool en) Mouse tracking. More...
void	**Imagine::endGraphics** () Terminate graphics application. More...
void	**Imagine::fillCircle** (int xc, int yc, int r, const AlphaColor &col, bool xorMode=false) Filled Circle. More...
void	**Imagine::fillCircle** (const IntPoint2 &c, int r, const AlphaColor &col, bool xorMode=false) Filled Circle (IntPoint2). More...
void	**Imagine::fillEllipse** (int x, int y, int w, int h, const AlphaColor &col, bool xorMode=false) Filled Ellipse. More...
void	**Imagine::fillEllipse** (const IntPoint2 &p, int w, int h, const AlphaColor &col, bool xorMode=false) Filled Ellipse (IntPoint2). More...

La documentation

Le site du cours → Installation Imagine++ → Instructions

```
void Imagine::fillCircle ( int                xc,
                           int                yc,
                           int                r,
                           const AlphaColor & col,
                           bool               xorMode = false
                         )
```

Fills a circle.

Parameters

xc,yc	center
r	radius
col	AlphaColor or Color
xorMode	XOR drawing (default=off). Used twice, recovers the original content

```
fillCircle(330,43,30,YELLOW);                                    // filled circle
```

Examples:

Graphics/test/example.cpp, and Graphics/test/test.cpp.

Plan de la séance

Déclaration, définition

Spécificités des tableaux

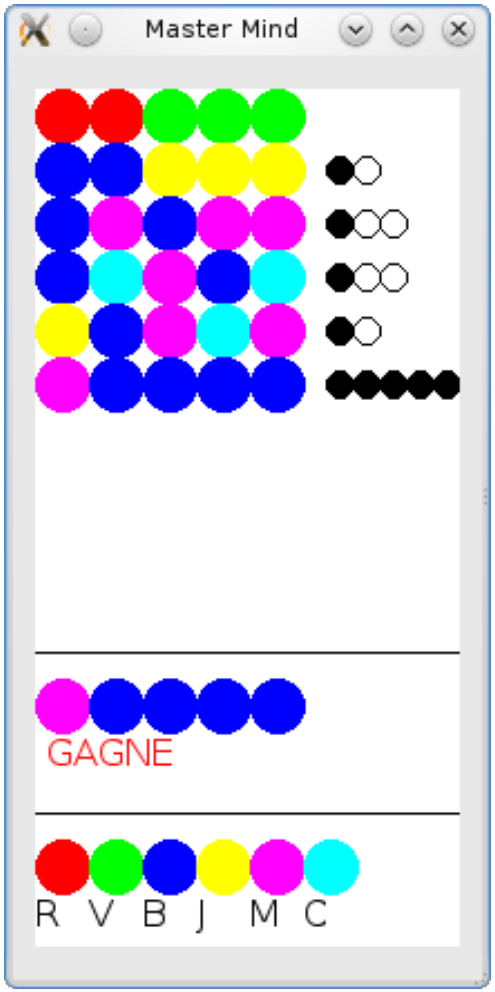
La librairie Imagine++

TP

Le TP du jour

Mastermind

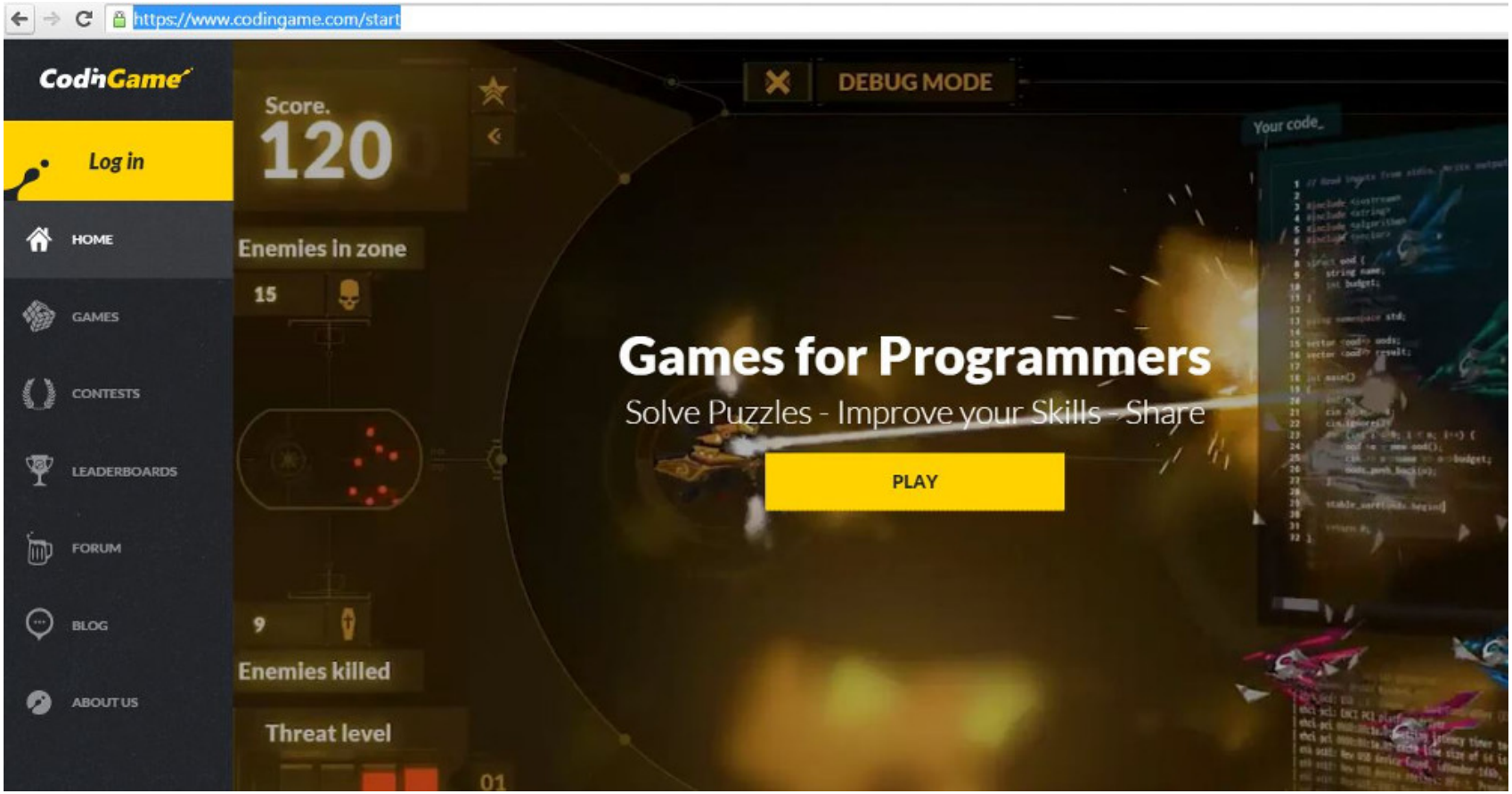
- Utilisation des tableaux
- Algorithmie
- Fonctions graphiques



Exercices

- Écrire un programme demandant à l'utilisateur d'entrer 10 entiers qui seront placés dans un tableau. Calculer et afficher le plus grand élément.
- Écrire une fonction qui inverse les valeurs d'un tableau (le premier élément devient le dernier, le second l'avant dernier...)
- Écrire une fonction prends deux tableaux, qui les compare terme à terme et qui renvoie `true` si tous les éléments sont égaux, `false` sinon.

CodinGame


https://www.codingame.com/start
CodinGame
Log in
HOME
GAMES
CONTESTS
LEADERBOARDS
FORUM
BLOG
ABOUT US
Score.
120
Enemies in zone
15
9
Enemies killed
Threat level
01
DEBUG MODE
Your code
Games for Programmers
Solve Puzzles - Improve your Skills - Share
PLAY