

Tema 2. Annex Punters i arrays en C++

Estructura de Dades

Grau en Enginyeria Informàtica Facultat de Matemàtiques i Informàtica, Universitat de Barcelona



Repàs



- Què és un punter?
 - Un punter és una variable que conté una adreça de memòria
- Declaració

```
<tipus> *<identificador>;
```

- Existeix una adreça especial que es representa mitjançant la constant nullptr
 - s'usa quan volem expressar que el punter no té adreça on apuntar



- Quan es declara un punter es reserva memòria per guardar una adreça de memòria
 - No és per emmagatzemar la dada a la que apunta el punter

L'operador &

- &<id> Retorna l'adreça de memòria on comença la variable <id>
- S'usa per assignar valors a les dades de tipus punter



L'operador *

- *<ptr> Retorna el contingut de l'objecte
 referenciat pel punter <ptr>
- S'usa per accedir als objectes als que apunta un punter

L'operador =

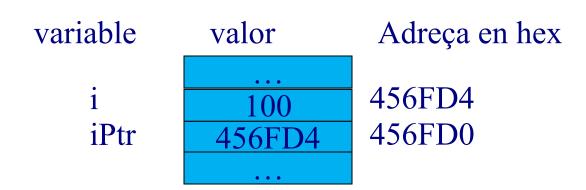
 A un punter se li pot assignar una adreça de memòria concreta, l'adreça d'una variable o el contingut d'un altre punter



Punters en C++: Exemple 1

```
int i;
int *iPtr; // un punter a un integer

iPtr = &i; // iPtr conté l'adreça de i
*iPtr = 100;
```





Punters en C++: Exemple 2

```
int main(){
 int y = 5, z = 3;
 int *nptr;
 int *mptr;
nptr = &y;
 z = *nptr;
 *nptr = 7;
mptr = nptr;
mptr = &z;
 *mptr = *nptr;
y = (*nptr) + 1;
 return 0
```



- Al campus virtual trobareu:
 - Un document anomenat Reforç Punters per repassar sobre el tema



Arrays en C++



- Hi ha diverses maneres de definir arrays en C++
 - De forma estàtica:

```
<tipus> <identificador>[<dimensio>];
int foo[5]; // defineix un array de 5 elements enters

foo[i] = 4; // accedeix a la casella i de l'array
foo[i]++; // incrementa en 1 el contingut de la casella i
```



- Hi ha diverses maneres de definir arrays en C++
 - De forma estàtica amb punters



- Hi ha diverses maneres de definir arrays en C++
 - De forma dinàmica amb punters



- Hi ha diverses maneres de definir arrays en C++
 - Amb STL vector

```
Cal fer: #include <vector>
std::vector<int> first; // empty vector of ints
O fer: using namespace std;
vector<int> first; // empty vector of ints
```

RECORREGUT

```
std::cout << "Print contents:";
for (std::vector<int>::iterator it = first.begin(); it !=
first.end(); ++it)
{
    std::cout << ' ' ' << *it; std::cout << '\n';
}</pre>
```



- Hi ha diverses maneres de definir arrays en C++
 - Amb STL vector
 - Té els seus propis mètodes (at(i), size(), empty(), front(), back(), push_front(e), push_back(e), ...
 - També es pot recórrer fent servir iteradors:

```
std::cout << "Print contents:";
for (std::vector<int>::iterator it = first.begin();
    it != first.end(); ++it)
{
    std::cout << ' ' ' << *it; std::cout << '\n';
}</pre>
```



Exemple

```
size t size = 10;
int sarray[10];
int *darray = new int[size];
// fer algun cosa amb ells:
for (int i=0; i<10; ++i) {
  sarray[i] = i;
  darray[i] = i;
// no oblideu esborrar darray quan acabeu
delete [] darray;
```



Exemple

```
#include <vector>
//...
size t size = 10;
std::vector<int> array(size); // guarda 10
enters
// fer algun cosa amb ell:
for(int i=0; i<size; ++i) {
  array[i] = i;
// no es necessari esborrar res
```