

5 min

LE BAGUETTE

PAR THIBAUT JUZEAU ET MARC CABOCHE



BAGUETTE EN QUELQUES NOMBRES

Point de vue code / implémentation :

- 0 warnings
- 1 variable globale
- 6 énumérations déclarées
- 11 types spécifiques
- 38 fonctions d'exécution
- + de 2 000 lignes de code (flex, bison, cpp)

Point de vue langage / machine à pile :

- 0 conflits
- 24 termes réservés
- 25 non-terminaux (+ 1 axiome)
- 36 terminaux

... et une infinité de possibilités !



LES MOTS CLÉS DE BAGUETTE

Programme		
TERMINER	//	/* */

Interactions		
ecrire	lire	pause

Variables (types)		
entier	reel	texte

Listes de variables		
liste	taille	supprimer

Tests logiques			
vrai	faux	ET	OU

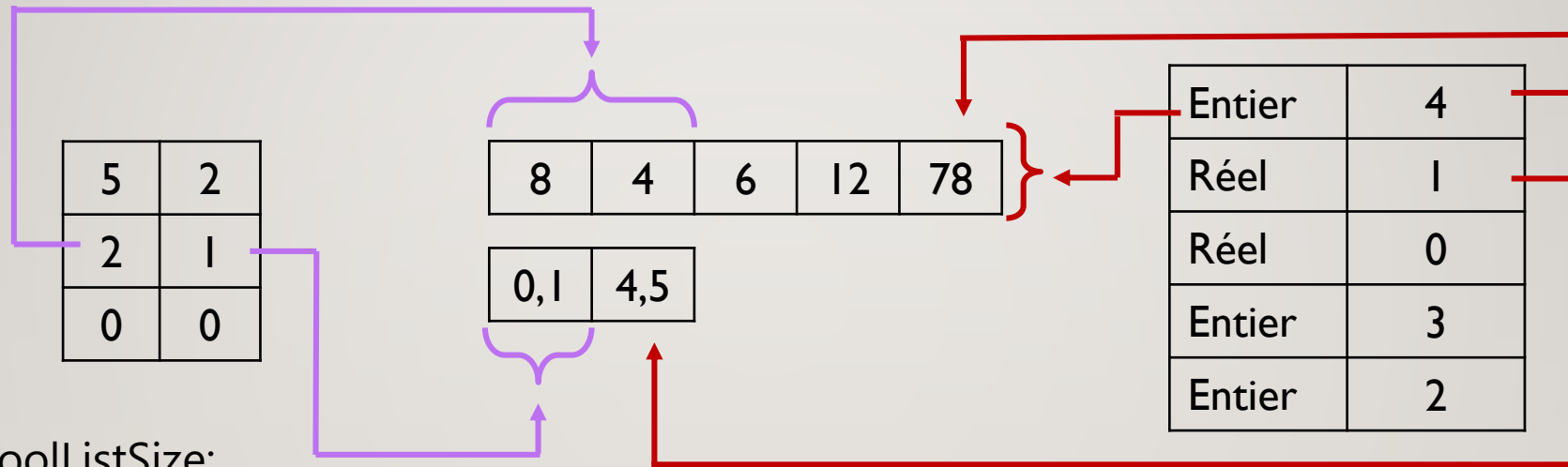
Conditions et boucles				
si	sinon	tantque	faire	pour

Fonctions		
vacant	recette	renvoyer

Zones mémoires	
ouvrir_zone_memoire	fermer_zone_memoire

LA GESTION MÉMOIRE AVEC BAGUETTE

I. Valeurs et zones mémoires



```
typedef struct {  
    unsigned int boolListSize;  
    unsigned int intListSize;  
    unsigned int doubleListSize;  
    unsigned int stringListSize;  
} memoryState;
```

```
stack<memoryState> memoryLayer;
```

```
deque<bool> boolList; //interne  
deque<int> intList;  
deque<double> doubleList;  
deque<string> stringList;
```

```
typedef struct {  
    valType type;  
    int tabPos;  
} valAccess;
```

```
stack<valAccess> executionPile;
```

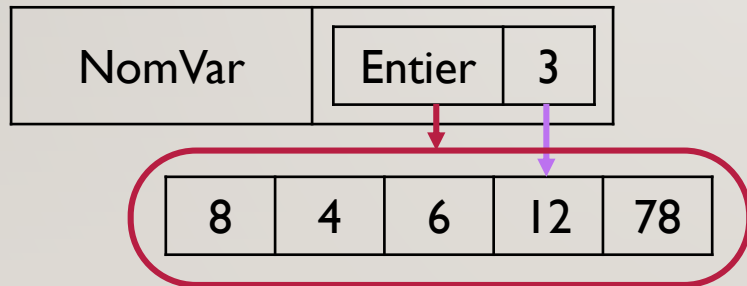
LA GESTION MÉMOIRE AVEC BAGUETTE

II. Variables et tableaux

Mêmes deque de valeurs et type
que précédemment

+

`map<string, valAccess> variables;`



```
deque<int> intArray;  
deque<double> doubleArray;  
deque<string> stringArray;
```

```
typedef struct {  
    unsigned int memoryLayer;  
    valType type;  
    deque<int> valuesPos;  
} tabAccess;
```

```
map<string, tabAccess> tableaux;
```



LA GESTION MÉMOIRE AVEC BAGUETTE

III. Fonctions et zones d'exécutions

```
typedef struct {  
    int reflnstruct;  
    valType returnType;  
    deque<param> listParam;  
} functionAccess;  
  
map<string, functionAccess> fonctions;
```

```
typedef struct {  
    string name;  
    unsigned int returnAdress;  
  
    map<string, valAccess> variables;  
    map<string, tabAccess> tableaux;  
} functionCall;  
  
stack<functionCall> currentExecution;
```

BAGUETTE : POUR QUI ?

- Langage simplifié mais complet
 - Langage pseudo-compilé
 - Syntaxe intuitive et en français
 - Gestion mémoire précise
 - Spécificités communes à d'autres langages
 - Inspiré du C++ et du PHP
- 
- Dev néophyte pour se familiariser avec la programmation (aussi bien logiciel que web)
 - Dev confirmé pour des projet rapides à moyennement complexe (pas de Framework)
 - Dev souhaitant gérer précisément la mémoire (systèmes embarqués de puissance moyenne)

NOUS RÉPONDONS À VOS QUESTIONS