Mémento Algorithmique-C++

Déclarations

Types Algorithmiques	Types C++ *(/nb octets)*
Entier	int*/4*, short*/2*, long*/4* (unsigned)
Réel	float*/4*, double*/8*
Caractère	char*/1*
Booléen	bool*/1*
Chaîne	string
Fichier	fstream
Tableau de *N type tab* // indices 1 à *N*	*type tab[N]* // indices 0 à *N*-1

Enregistrement
type1 champ1
...
fin enregistrement
	struct {
	type1 champ1 ;
	...
	}

Pointeur vers *type p*	*type * p*
Définition d'un alias de type Type *T* = *XXX*	**Définition d'un alias de type** typedef *XXX T* ;

Opérations

Arithmétiques	+ - / * div mod	+ - / * / %
Logiques	∧(et) ∨(ou) ¬(non)	&&(and) ||(or) !(not)
Affectation	←	=
Comparaisons	= ≠ < ≤ > ≥	== != < <= > >=
Entrées/Sorties	lire *x* lire *x* dans *f* écrire *d* écrire *d* dans *f*	cin >> *x* *f* >> *x* cout << d << endl *f* << *d*
Fichiers	lecture *f*, *n* écriture *f*, *n* fermer *f* fini *f*	*f*.open(*n*,ios::in) *f*.open(*n*,ios::out|ios::trunc) *f*.close() *f*.eof()
Chaînes	*s:i* |*s*| *s~t*	*s*.at(*i-1*) *s*.length() *s+t*
Tableaux	*t*[*i*] |*t*|	*t*[*i-1*] *pas d'équivalent*
Mémoire	adr(*x*) mém(*p*) *p* ← allocation *type* désallouer *p*	&*x* **p* *p* = new *type* delete *p* delete[] *p* *(si tableau)*

Conditionnelles

si *Condition* alors
Séquence1
 	
sinon
Séquence2
fin si
	if (*Condition*) {
	Séquence1
	} // if
	else {
	Séquence2
	} // else

selon *Expression* dans
Valeur1 : *Séquence1*
...
A défaut : *Séquence*
fin selon
	switch (*Expression*) {
	case *Valeur1* : *Séquence1* ; break ;
	...
	default : *Séquence* ;
	} // switch

Répétitives

tant que *Condition* faire
Séquence
fin tant que
	while (*Condition*) {
	Séquence
	} // while

pour *Var1* de *Expression1* à *Expression2*
faire
Séquence
fin pour
	Var2 = *Expression2* ;
	for (*Var1* = *Expression1*; *Var1* <= *Var2*;
	++*Var1*) {
	Séquence
	} // for
	Var1 = *Var2*;

répéter *Séquence* jusqu'à *Condition*	do { *Séquence* } while (!*Condition*);

Routines

fonction *fun* (Tableau d'Entiers *tab*) : Réel procédure *pro* (Booléen *b*, Caractère réf c)	double *fun* (int *tab[]*) void *pro* (bool *b*, char & *c*)