ALGORITHME.	C++
• Commentaire ligne • Commentaires sur plusieurs lignes	• // ″commentaire ligne″ • /* ″commentaire plusieurs lignes ″ */
• constante entier N=10 ; • constante entier non signé N=10 ; • constante entier long N=10 ; • constante réel M=.05 ; • constante caractere N= « Veuillez… » ;	• const N=10 ; (ça cera une const int par défaut) • const unsigned N=10 ; • const long int N=10 ; • const float M=.05 ; • const char *N= « Veuillez… »;
♦ var • entier x ; • entier non signé x ; • entier long x ; • entier long non signé x ; • réel y ; • réel double y ; • caractère c ; (un caractère) • caractère nom[20] (20 caractèresmax); • chaine nom ; • bool z ; (faux, vrai ou 0, 1)	• int x ; • unsigned int x ; • long int x ; • unsigned long int x ; • float y ; • double y ; • char c ; • char nom[20] ; • string nom ;[s'il y a #include<cstring.h>] • bool z ;

Entree sortie
Début /*PP*/
 	
Fin
	#include <iostream.h>
	int main()
	{
	
	return 0 ;
	(*ou* getch () ; [si il y'a *#include <conio.h>*])
	}

• x ← 4 ;	• x=4 ;
• y ← x+1.26 ;	• y=x+1.26 ;
• afficher « Donner… » ; • afficher «Donner… » ; [et revenir à la ligne] • afficher «La somme est \n »,D ; [et revenir à la ligne] • afficher « La somme est », D ; [et revenir à la ligne] • afficher «Donner… »,D ,«cogner… » ;	• cout << ″ Donner… ″ ; • cout << ″ Donner… \n ″ ; • cout << ″ La somme est \n ″ << D ; (l'affichage de la valeur de D **n'est pas** sur la meme ligne que la phrase) • cout << ″ La somme est ″ << D << endl ; (l'affichage de la valeur de D **est** sur la meme ligne que la phrase) • cout<< ″ Donner… ″ <<D<< ″ cogner… ″ ;
• saisir C ;	• cin >> C ;
• afficher aux coordonnées indiqués : Cadrer (x,y) ; Afficher ……..	• goto(1,2) ; instructions ;
• Tout Effacer Ecran ;	• clrscr () ;
• Pause ;	• getch() ;
• Stopper ;	• break ;

• Si (condition) alors
Instruction 1 ;
Instruction n ;
fsi ; (; non obligatoire après fsi)
• Si (condition) alors
Instruction 1 ;
Instruction n ;
 	
Sinon
Instruction 1 ;
Instruction n ;
fsi ;
• Si (Qt>0 et Qt<30) alors
……
	• if (condition)
	{ instruction 1 ;
	instruction n ;
	}
	• if (condition)
	{ instruction 1 ;
	instruction n ;
	}
	else
	{ instruction 1 ;
	instruction n ;
	}
	• if ((Qt>0) && (Qt<30))
	{ ……

• =	• ==

• ≠ • < • > • ≤ • ≥	• != • < • > • <= • >=
• Pour i ← 1 jusqu'à 10 faire Instruction 1 ; Instruction n ; Fpour ; • Répéter Instruction 1 ; Instruction n ; i ← i + 1 ; tant que (i <=10) ; • Tant que (i <10) faire Instruction 1 ; Instruction n ; i ← i + 1 ; ftant que ;	• for (i=1 ; i<=10 ; i++) [ici i++ veut dire i=i+1] { Instruction 1 ; Instruction n ; } • do { instruction 1 ; instruction n ; i=i+1 ; } while (i<=10) ; • while (i<10) { instruction 1 ; instruction n ; i=i+1 ; }

• Cas (choix) dans
 	
v1 : début ;
instruction 1 ;
instruction n ;
fin ;
v2 : début ;
instruction 1 ;
instruction n ;
fin ;
autres : instruction 1 ;
instruction n ;
 	
fin ;
 	
fcas ;
	• switch (choix)
	{
	case 1 : { instruction 1 ;
	instruction n ;
	break ;
	}
	case 2 : { instruction 1 ;
	instruction n ;
	break;
	}
	default : { instruction 1 ;
	instruction n ;
	break ;
	}
	
	}

• Ent T[10] ; (tableau à une dimension) • Ent T[10][15] ; (tableau à deux dimensions)	• int T [10] ; • int T[10][15] ;
• ent T[7]={1,2,5,9,4,2,4} ; • chaine T[3]={"Bonjour", "Sava","Toi"} ;	• int T[7]={1,2,5,9,4,2,4}; • string T[3]={"Bonjour", "Sava","Toi" } ;

• STRUCTURE
• Type nomstructure = structure
Chaine nom ;
Ent age ;
Caractere sexe ;
Finstructure ;
ici le nomstructure devient un type.
 	
• Declaration de la structure :
Nomstructure lettre ;
 	
• Utilisation d'un champ :
Saisir nom de la structure ;
Afficher age 1 de la structure ;
 	
• Structure avec tableau :
Il faut la declarer tel un tableau :
Nomstructure lettre[10] ;
 	
• Utilisation d'une structure avec tableau :
Saisir nom de la structure 1 ;
	• En C : typedef struct {
	string nom ;
	int age[12] ;
	char sexe ;
	} eleve ;
	En C ++: struct eleve{
	string nom ;
	int age[12];
	char sexe;
	};
	• eleve el ;
	
	• cin >> el.nom ;
	cout<<el.age[1] ;
	
	• eleve el[10] ;
	
	• cin >> el[1].nom ;

Afficher age 1de la structure1 ;	cout << el[1].age[1]

• PROCEDURE
• declaration procédure remplir
• appel procédure remplir :
remplir(T,taille) ;
 	
• procédure remplir (par adresse T[], par valeur n)
début
instruction 1 ;
instruction n ;
fin ;
	• void remplir() ;
	• remplir() ;
	
	• void remplir() ;
	{
	instruction 1 ;
	instruction n ;
	}

• declaration FONCTION remplir
• appel fonction remplir :
x ← remplir(T,taille) ;
• fonction remplir (par valeur T[], par valeur n) : le type de la valeur à retourner (ex :ent)
début
instruction 1 ;
instruction n ;
retourne (y) ;
fin ;
	• int remplir () ;
	• x=remplir () ;
	
	• int remplir()
	{
	instruction 1 ;
	instruction n ;
	return (y) ;
	}

• FICHIER * vehicule ; (declaration du fichier vehicule)	• FILE * vehicule ;
• OUVERTURE d'un FICHIER et sa FERMETURE Vehicule ← ouvrir (″vehicule.dat″, écriture) ; Instructions 1 ; Instructions n ; Fermer (vehicule);	• Vehicule= fopen (″vehicule.dat″, ″w″) ; Instruction 1 ; Instructions n ; fclose (vehicule) ;
• ECRITURE Ecrire (&nom logique, veh) ;	• fwrite(&veh, sizeof(veh), 1, vehicule) ;
• LECTURE simple Lire (nom logique, veh) ;	• fread(&veh, sizeof(veh), 1, vehicule) ;

• LECTURE multiple
Lire (nom logique, veh) ;
Tant que nom fin de fichier(nom logique) faire
Instruction 1 ;
Instruction n ;
Lire (nom logique, veh);
Ftant que ;
	• fread(&veh, sizeof(veh), 1, vehicule) ;
	while (!feof(vehicule)) ;
	{
	instruction 1 ;
	instruction n ;
	fread(&veh, sizeof(veh), 1, vehicule) ;
	}

• POSITIONNEMENT Positionner (nom logique, numéro enregistrement) ; Instructions ;	• fseek(vehicule, (long)(num enr – 1) * sizeof(veh) SEEK-SET) ; Instructions ;
•	•
•	•
•	•
•	•
•	•
•	•