FICHE TECHNIQUE JAVA

Commenter une ligne	// commentaire
Commenter plusieurs lignes	/* commentaire */

- **Types**

Booléen	Boolean
Entier	int, short, long
Reel	double
Chaine de caractères	String

- **Les variables**

Déclaration d'une variable	int nomVariable ;
Affectation d'une valeur	nomVariable = valeur ;

- **Affichage et Saisie**
  Attention au respect des minuscules et majuscules.

Librairie	import java.util.Scanner ;
Affichage d'un message	System.out.println (« Message à afficher ») ;
Affichage d'une variable	System.out.println(nomVar);
Affichage d'un message et d'une variable	System.out.println (« Message »+nomvar) ;
Saisie d'une valeur	Utilisation de la classe Scanner Scanner sc = new Scanner(System.in); String str = sc.nextLine(); int i = sc.nextInt();

- **Les conditions**

Égalité	a == b
Différence	a != b
Inférieur	a<b Ou a<=b
Supérieur	a>b Ou a>=b
Combiner les conditions	ET logique : && OU logique : ||

- **Les structures de contrôle**

La structure itérative	if (condition évaluée sous forme booléenne)
	{
	// Traitement si vrai
	}
	else
	{
	//Traitement si faux
	}
	
	switch (élément à analyser)
	{
	case valeur1 :
	//traitement si élément = valeur1
	break ;
	case valeur2 :
	// traitement si élément =valeur2
	break ;
	default :
	// traitement si élément différent des valeurs
	
	}

La structure tant que	while (condition évaluée sous forme booléenne)
	{
	// Traitement à répéter
	}

La structure pour	for (compteur = valDépart ; compteur <= valFin ;compteur++)
	{
	// Traitement à répéter
	}

- **Les tableaux**

Déclaration et allocation mémoire	int[] tableau = new int[50];
Initialisation d'un tableau	int tableau[] = {10,20,30,40,50};
Parcours d'un tableau	for (int i = 0; i < tableau.length ; i ++) { ... }

- **Procédures et fonctions**

Création d'une procédure	public void nomProc(typeparam1 nomparam1, typeparam2 nomparam2)
	{
	//programme
	}

Appel d'une procédure	nomProc (3.5, 10) ;

Création d'une fonction	public typeRetourné nomFonc(typeparam1 nomparam1, typeparam2 nomparam2)
	{
	//Programme
	return valeur;
	}

Appel d'une fonction	nomVar = nomFonc ("chaine", 2) ;

- **Création des classes**

Création d'une classe	public classe NomClasse {
	//attributs
	//constructeur
	//méthodes
	}

Visibilité des éléments de la classe	private protected public
Attribut	int attribut ;
Constante	final type nomconstante = valeurnommodifiable ;
Attribut de classe	static type attributstatic = valeurpartagee;

Constructeur	public NomClasse (typeparam param)
	{
	//initialisation des attributs de la classe
	}

Méthode	public void nomMethode()
	{
	//corps de la méthode
	}

Méthode de classe	static void nomMethode()
	{
	//corps de la méthode
	}

Classe abstraite	abstract class NomClasseAbstraite {}

Méthode abstraite	abstract void nomMethode()
	{
	//corps de la méthode
	}

Héritage d'une classe	public class NomClasseFille extends NomClasseMere {}

- **Utilisation des classes**

Création d'un objet	NomClasse objet; // déclaration de l'objet
	objet = new NomClasse(); // création de l'objet
	
	OU NomClasse objet = new NomClasse();
	// déclaration et création de l'objet

Appel d'une méthode	nomObjet.nomMethode() ;
Appel d'une méthode de classe	NomClasse.nomMethode() ;