**Informations élémentaires manipulées par un algo**

Algorithme : Info0101

Déclaration :

Variables :

N : entier

Début :

N🡨5

Ecrire ("Vive info0101 (et ses " +n+ "professeurs)")

Fin

**Affectation**

Algo : échange

Déclaration :

Variables :

A, b, tmp : entier

Début :

|  |  |
| --- | --- |
| A🡨2  B🡨3  Tmp🡨a  A🡨b  B🡨tmp | A🡨2  B🡨3  A🡨a+b  B🡨a-b  A🡨a-b |

Fin

**Lecture (demande à l'utilisateur)**

Var2🡨lire ("entrez une valeur")

Var1🡨lire ()

**Ecriture (affichage)**

Ecrire ("je parle")

**Nombre aléatoire**

Var🡨 aleatoir () 🡪entre [0 ; 1[

Var 🡨aleatoir () \* 10-5 🡪entre [-5 ; 5[

**Commentaires :**

//

/\*…\*/

**Opérateurs arithmétiques**

'+' , '-' , '/' , '\*'

Règles, comme en vrai, '\*' et '/' d’abord puis '+' et '-'

**Opérateurs rationnels**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| < | <= | > | >= | == | != |
| Inférieur | Inf ou égal | Supérieur | Sup ou égal | Egalé | Différent |

**Opérateurs logiques**

Retournent un booléen

**ET** vrai ssi les 2 opérantes sont vrai

**OU** (ou inclusif), vrai si au moins un des deux opérateurs est vrai

**NON** donne vrai si c’est faux et faux si c’est vrai

**Instructions structurées**

Contient des instructions de bases

Structures de sélection (conditionnelles) avec opérateurs rationnels

**Si… alors …. Sinon**

Si conditionVraie, alors

Action

Sinon

Action

FinSi

Exemple :

3) calcul du salaire d’un fonctionnaire

Algorithme :

Déclaration

Variables

TauxHoraire, salaire, prime : (réel)

NbHeure : (entier)

Début :

Ecrire (" entrez le taux horaire : ")

TauxHoraire 🡨 lire ()

Ecrire (‘’ entrez le nombre d’heure de travail :’’)

NbHeure 🡨 lire ()

Si nbHeure > 100 alors

Écrire (‘’ entrez la prime :’’)

Prime🡨lire ()

Sinon

Écrire (‘’ entrez la prime de sous emploie’’)

Prime 🡨lire ()

FinSi

Salaire🡨nbHeure \* tauxHoraire + prime

Écrire ("le salaire est"+ salaire +"€")

Fin

4) maximum de 2 valeurs

Algo :

Déclaration :

Variable :

A, b, max (réel)

Début :

A🡨 lire (‘’entrez la première valeur :’’)

B🡨lire (‘’entrez la première valeur :’’)

Si a>b alors

Max🡨a

Sinon

Max🡨b

FinSi

Écrire (‘’ le maximum est ‘’+max)

Fin

**Cas…parmi**

Exprime de façon plus courte de la conditionnelle

Syntaxe

Cas expression parmi

Valeur : action

Valeur : action

Fincas

Exemple

5) prix d’un billet de train après réduction

Algo :

Déclaration :

Variables :

Réduc : (réel)

Prix : (réel)

Voyageur : (caractère)

Début :

Prix🡨lire (‘’entrez le prix du billet’’)

Voyageur🡨lire (‘’ entrez le type de voyageur ‘’)

Cas voyageur parmi

‘a’ : réduction🡨50

‘e’ : réduc🡨100

‘v’ : réduc🡨60

Défaut : réduction🡨 0

FinCas

Prix🡨prix-(prix\*(réduc/100))

Écrire (prix)

Fin

**Trace d’exécution d’un algorithme**

Permet de suivre pas à pas l’algo (tableau)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Inst | Contrôle | Variable | | | | | Locales | | Affichages |
| Avant | Avant | NbHeure | /tauxHoraire | SalaireBrut/ | Prime | /primeSE | Résultat |  | |
| 1 |  | ? | ? | ? | ? | ? |  | | TauxHoraire ? |
| 2 |  |  | 10 |  |  |  |  | | NbHeure |
| 3 |  | 150 |  |  |  |  |  | |  |
| 4 |  |  |  | 1500 |  |  |  | |  |
| 5 | NbHeure>100 ? vrai |  |  |  |  |  |  | |  |
| 5a.1 |  |  |  |  |  |  |  | | Prime ? |
| 5a.2 |  |  |  |  | 500 |  |  | |  |
| 5a.3 |  |  |  | 2000 |  |  |  | |  |
| 6 |  |  |  |  |  |  |  | | Salaire :2000 |
| Fin |  |  |  |  |  |  |  | |  |

**Exercices :**

Exercice 1 :

1)

Algorithme : entier constante

Déclaration :

Constantes :

K=20

Var :

Val : entier

Début

Val🡨lire ("entrez un entier :")

Val🡨val\*k

Ecrire(val)

Fin

2)

Algo : affiche

Déclaration :

Var :

K : entier

Début

K🡨lire ("entrez une valeur entre 1 et 3")

Si k==1 alors

Ecrire ("un")

Sinon

Si k==2 alors

Ecrire ("deux")

Sinon

Ecrire ("trois")

FinSi

FinSi

Fin

3)

Algo : égalités réelles

Déclaration :

Var :

Val1, val2: réels

Début

Val1🡨lire ("entrez le premier réel :")

Val2🡨lire ("entrez le deuxième réel :")

Si val1==val2 alors

Ecrire("oui")

Sinon

Ecrire ("non")

FinSi

Fin

4)

Algo : affiche

Déclaration :

Var :

K : entier

Début

K🡨lire ("entrez une valeur entre 1 et 3:")

Cas k parmi

1 : écrire ("un")

2 : écrire ("deux")

3 : écrire ("trois")

Défaut : écrire ("saisi incorrecte")

Fincas

Fin

Exercice 3 :

Algo : ?

Déclaration :

Variable :

Montant, taux=1.46 : entier

Devise : char

Début

Écrire ("montant :")

Lire🡨 montant

Écrire :(‘’devise : euro (e) ou dollar canadien(d) :")

Lire🡨devise

Si devise==’d’alors

Montant=montant/taux

Écrire :( "montant en euro :"+ montant)

Sinon

Si devise==’e’ alors

Montant=montant\*taux

Écrire :( "montant en dollar canadien :"+ montant)

Sinon

Ecrire (« erreur »)

FinSi

FinSi

Fin

Exercice 4 :

Algo :

Déclaration :

Variable :

Somme, reste, nb : entier

Début :

Somme🡨 lire ("entrez une somme entière :")

Reste🡨montant%500

Nb🡨(montant-reste) /500

Même code pour le reste

Montant 🡨 reste

Ecrire ("billet de 500 : "+nb)

Exercice 5 :

Algo :

Décla :

Variable :

Nb1, nb2 : réel

Début

Nb1🡨lire (----)

Nb2🡨lire (----)

Si nb1==0 ou nb2==0 alors

//manque passage

Sinon

Si (nb>0 et nb2>0) ou (nb1<0 et nb2<0) alors

Ecrire ("le résultat est positif")

Sinon

Ecrire ("le résultat est 0")

FinSi

FinSi

Fin

**Algo :**

Structures itératives (boucles):

**TantQue** condition faire

Action

FinTantQue

**Pour** variable\_de\_controle allant de val1 à val2 faire

Action

FinPour

**Faire**

Action

Tanque condition (la condition doit être fausse pour sortir de la boucle)

Algo : Maximum

Décla :

Var :

Val, max : entier

Début :

Val🡨lire ("Entrez une valeur : ")

Max🡨val

TantQue val ! = 0 Faire

Val🡨 lire ("----")

Si val> max Alors

Max 🡨val

FinSi

FinTantQue

Ecrire 🡨 ("le max est " + max)

Fin

Tableau d'exécution de Maximum :

|  |  |  |  |
| --- | --- | --- | --- |
| Instruction | Contrôle | Variables locales | |
| Val | Max |
| Avant | - | ? | ? |
| 1 | "entrez val" | 3 |  |
| 2 |  |  | 3 |
| 3 | Val ! = 0 ? Vrai |  |  |
| 3.1 | "Entrez val" | 4 |  |
| 3.2 | Val>max |  |  |
| 3.2.a.1 |  |  | 4 |
| … |  |  |  |

Algo : PGCD

Décla :

Var :

A, b : entier

Début :

A🡨 lire ("---")

B🡨 lire ("---")

TantQue a ! = b Faire

Si a>b Alors

A🡨a-b

Sinon

B🡨b-a

FinSi

FinTantQue

Ecrire (" le PGCD est " +a)

Fin

Algo :

Décla :

Var :

Val, max : entier

Début :

Val 🡨 lire ("---")

Max 🡨 val

Pour : i allant de 1 à 3 Faire

Val 🡨 lire ("---")

Si val > max Alors

Max🡨val

FinSi

FinPour

Ecrire (max)

Fin

Algo : Saisie

Décla :

Var :

Nb : entier

Début :

Faire

Nb🡨 lire ("Entrez une valeur supérieure à 10 : ")

TantQue nb<10

Ecrire (nb)

Fin

**Cours 2 : le langage java**

Etapes de prog : écrire, compiler et exécuter

**Premier programme java :**

Une classe = un programme

Class PremierProg {

Public static void main (String [] args) {

System.out.println(« Mon premier programme Java (et non le dernier !) ») :

}

}

Mots clés :

-public, class, static, void, main, String

-Les {}

-PremierProg : nom du programme

-Public static void main (String [] args) : procédure principale

**Affichage console :**

System.out.print(« … ») ou System.out.println(« … ») : pour afficher et le 2 avec retour a le ligne

Entrées au clavier : importation de la classe Scanner

Déclaration d’une variable de type Scanner : Scanner clavier= new Scanner(System.ln)

Lecture d’une entrée d’un format donné :

…

**Conventions :**

**Identificateur :** suite de caractères, composé de lettre, chiffre et « \_ », doit commencer par une lettre, distinguer majuscules et minuscules

**Programmer en Java :**

Programmer à la maison :

-JDK + éditeur de texte

Types de données en JAVA :

-entiers :

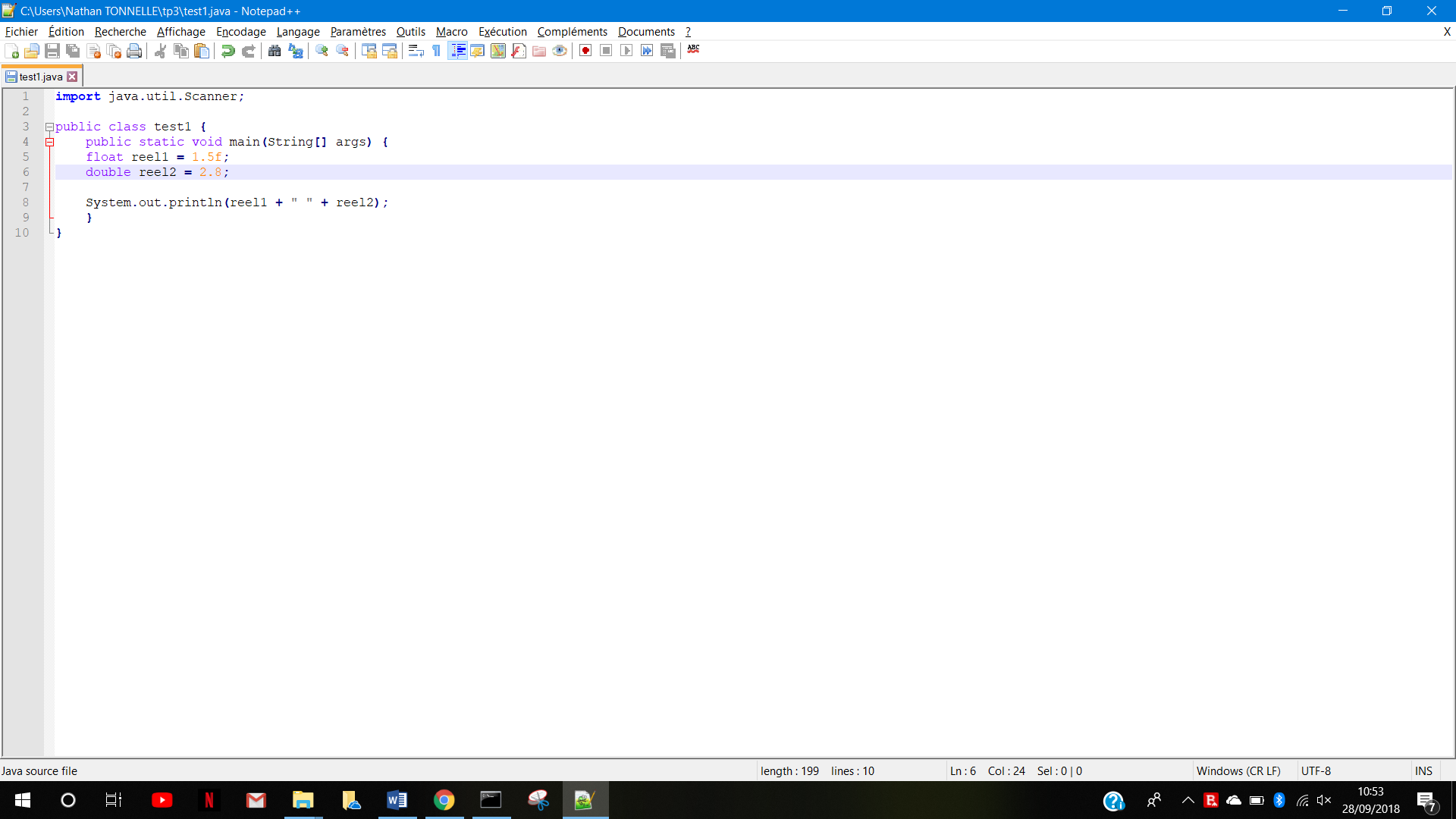
Byte [-128 ; 127]

Short [-32768 ; 32767]

Int [-2147483648 ; 2147483647]

Long [-9.22e+18 ; 9.22e+18]

-flottant :

Float mettre apres le chiffre à virgule un f pour que java prenne le chiffre comme un float et pas un double

Double

-caractères

Ils sont représentés en Unicode

Char 'a' '+' 'E'

-booléen :

Boolean (true, false)

Initialisation de variable :

Ou int n=15 ;

Int n ;

N=15 ;

Pour une constante :

Final int n=20 ; 🡪 n= constant

Opérateur en Java :

Opérateurs arithmétiques :

Binaire : + - / \* %

Unaire : -(opposé) +(identifié)

Opérateur relationnels (logique) :

Evaluation de la partie gauche d'abord si vrai, côté droit sinon passe

|  |  |
| --- | --- |
| ! | Négation |
| & | Et |
| ^ | "ou" exclusif (un des 2 doit être vrai) |
| | | "ou" inclusif (les 2 doivent être vrai) |
| && | "et" avec court-circuit |
| || | "ou" inclusif avec court-circuit |

Opérateur d'affectation :

= i=32 x=i+8

Opérateur d'incrémentation, de décrémentation

++ (+1) -- (-1)

A gauche pré-incrémentation

A droite post-incrémentation

Opérateur de cast (transtypage) :

* Force la conversion

(Type) expression

(Double) (10/3) 🡪🡪 3.0

(Double) 10/3 🡪🡪 3.333333

**Premières structures de sélection en Java :**

Blocs et instruction :

-instruction simple fini par ";"

Structurée fini par "}"

-blocs : suite d'instruction entre {}

{} : vide, 1 ou plusieurs instructions

If…else (si… alors … sinon) :

If (condition) {

Instruction

}

Else {

Instruction

}

Switch (cas… parmi) :

Switch (expression) {

Case constante 1 :

Instruction

Break ;

Défaut :

Instruction

Break ;

}

**STRUCTURES ITÉRATIVES EN JAVA :**

For (pour)

For (initialisation ; condition ; incrémentation) {

Instruction

}

While (TantQue)

While (condition) {

Instruction

}

Do… while (faire… TantQue)

Do {

Instruction

} while (condition) ;

TD n\*2 Boucles

Exercice 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Algo 1 | | | | | |
| Instruction | Contrôle | Variables locales | | | |
| N | P | I | Résultat |
| Avant | - | 5 | ? | ? | ? |
| 1 |  |  | 1 |  |  |
| 2 |  |  |  | 5 |  |
| 3 | I ≠ 0 ? Vrai |  |  |  |  |
| 3.1 |  |  | 5 |  |  |
| 3.2 |  |  |  | 4 |  |
| … |  |  |  |  |  |
| 4 |  |  |  |  | P |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Algo 2 | | | | | | |
| Instruction | Contrôle | Variables locales | | | | |
| N | X | P | I | Résultat |
| Avant | - | 5 | 2 | ? | ? | ? |
| 1 |  |  |  | 1 |  |  |
| 2 | I allant de 1 à n |  |  |  |  |  |
| 2.1.1 |  |  |  | 2 | 1 |  |
| 2.1.2 |  |  |  | 4 | 2 |  |
| 2.1.3 |  |  |  | 8 | 3 |  |
| … |  |  |  |  |  |  |
| 3 |  |  |  |  |  | P |

Exercice 2

Algorithme : PGCD euclide

Variables locales :

A,B,r : entier

Début

A🡨lire()

B🡨lire()

r=a%b

tantque r≠0 faire

a=b

b=r

Fin tantque

ecrire🡨(b)

Fin

Exercice 3

1)

78%2=0 78/2=39

39%2=1 39/2=19

…

Arrêt à 0

Algo : BinFixe

Décla :

Var :

Val, i : entier

Constante :

nbBits=12

Début :

Val🡨lire ("—")

Pour i allant de 1 à nbBits faire

Ecrire (val%2)

Val🡨val/2

FinPour

Fin

2)

Algo : BinVar

Décla :

Var :

Val : entier

Début :

Val 🡨 lire ("---")

TantQue val ! = 0 Faire

Ecrire (val%2)

Val 🡨val/2

FinTantQue

Fin

Exercice 4 :

1)

Algo :

Décla :

Var :

X, n, abs-n, résultat : entier

Début :

X🡨 lire ("---")

N🡨 lire ("---")

Si n>0 alors

Abs-n 🡨n

Sinon

Abs-n🡨-n

FinSi

Résultat 🡨1

Pour i allant de 1 à abs-n Faire

Résultat 🡨résultat\*x

FinPour

Si n<0 alors

Résultat 🡨1/résultat

FinSi

Ecrire (résultat)

Fin

3)

X, x2 : entier

Début :

X🡨lire ("---")

X2🡨x\*x

X🡨x2\*x2

X🡨x\*x

X🡨x\*x2

Ecrire (x)

Fin

**Cours 3 : fonctions et procédures**

Algo : maximum

Déclaration :

Var :

A, b, c, d, maxAB, maxCD : entier

Début :

A 🡨 lire ("—")

B 🡨 lire ("—")

C 🡨 lire ("—")

D 🡨 lire ("—")

Si a≥b alors

maxAB🡨a

Sinon

maxAB🡨b

FinSi

Si c≥d alors

maxCD🡨C

Sinon

maxCD🡨D

FinSi

Ecrire (maxAB "+" maxCD)

Fin

\*Algo : max

Décla :

Var locales :

A, b, c, d, maxAb, maxCD : entier

Début :

a🡨 lire ("—")

b🡨 lire ("—")

c🡨 lire ("—")

d🡨 lire ("—")

maxAB 🡨 max(a,b)

maxCD🡨 max(c,d)

Ecrire (maxAB "+" maxCD)

Fin

\*Fonction max (x : entier, y : entier) : entier

Déclaration :

Var locales :

N : entier

Début :

Si x>y alors

N🡨x

Sinon

N🡨y

FinSi

Retourner n

Fin

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| \*Algo | | | | | | | | |
| Instruction | Contrôle | Var locales | | | | | | Écran |
| A | B | C | D | maxAB | maxCD |
| Avant |  | ? | ? | ? | ? | ? | ? |  |
| 1 |  | 1 | 2 | 4 | 3 |  |  |  |
| 2 | Max(a,b) |  |  |  |  | 2 |  |  |
| 3 | Max(c,d) |  |  |  |  |  | 4 |  |
| 4 |  |  |  |  |  |  |  | 2 + 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| \*Fonction | | | | | |
| Instruction | Contrôle | Variables locales | | | Renvoie |
| X | Y | N |
| Appel ° 1 | | | | | |
| Avant |  | 1 | 2 | ? |  |
| 1.b.1 |  |  |  | 2 |  |
| 2 |  |  |  |  | 2 |
| Appel ° 2 | | | | | |
| Avant |  | 4 | 3 |  |  |
| 1.a.1 |  |  |  | 4 |  |
| 2 |  |  |  |  | 4 |

TD n°3 les fonctions :

Exercice 1 :

1)

|  |  |  |  |
| --- | --- | --- | --- |
| Algo 1 | | | |
| Instruction | Contrôle | Variables locales | |
| X | Y |
| Avant |  | ? | ? |
| M1 |  | 1 |  |
| M2 |  |  | 2 |
| M3 | F (3, 2) | -1 |  |
| M4 | -1 2 |  |  |
| M5 | F (1, 2) |  | 3 |
| M6 | -1 3 |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fonction 1 | | | | | |
| Inst | Contrôle | Variables locales | | | |
| U | V | X | Résultat |
| F (3 , 2) |  | 3 | 2 |  |  |
| F1 |  |  |  | =2\*3+1=7 |  |
| F2 |  |  |  |  |  |
| F2a.1 |  |  |  |  | =2-3=-1 |
|  |  |  |  |  |  |
| F (1 , 2) |  | 1 | 2 |  |  |
| F1 |  |  |  | =2\*1+1=3 |  |
| F2 |  |  |  |  |  |
| F2b.1 |  |  |  |  | =2+1=3 |

|  |  |  |  |
| --- | --- | --- | --- |
| Algo 2 | | | |
| Instruction | Contrôle | Variables locales | |
| I | G |
| Avant |  | ? | ? |
| M1 |  | 2 |  |
| M2 | 2 9 |  |  |
|  | | | |
| Fonction 2 | | | |
|  |  | J | G |
| Avant |  | 2 |  |
| G1 |  | =2+2=4 |  |
| G2 | G : 4 |  |  |
| G3 |  |  | (f(4)+4)=5+4=9 |
|  |  |  |  |
|  |  | I | F |
| Avant |  | 4 |  |
| F1 |  | 5 |  |
| F2 | F : 5 |  |  |
| F3 |  |  | 5 |

Exercice 2 :

1)

Fonction : estBisextile (année : entier) : booléen

Déclaration :

Variables locales :

Résultat : booléen

Début :

Si (année % 400==0) ou (année % 4==0 et année %100 ! = 0) alors

Résultat 🡨vrai

Sinon

Résultat🡨 faux

FinSi

Retourner : résultat

Fin

2)

Fonction : nombreJour (mois : entier, année : entier) : entier

Déclaration :

Variables locales :

Résultat : entier

Début :

Si mois ==1 ou mois ==3 ou mois ==5 ou mois==7 ou mois == 8 ou mois ==10 ou mois==12 Alors

Résultat 🡨 31

Sinon

Si mois ! = 2 Alors

Résultat 🡨30

Sinon

Si estBisextile (année) alors

Résultat🡨29

Sinon

Résultat🡨28

FinSi

FinSi

FinSi

Retourner résultat

Fin

3)

Fonction : estValide (jour : entier, mois : entier, année : entier): booléen

Déclaration :

Variables locales :

Début :

Retourner mois>=1 et mois<12 et jour >=1 et jour<=nombreJour (moi, année)

Fin

4)

Algorithme :

Déclaration :

Variables locales :

Jour, mois, année, jourL, moisL, annéeL : entier

Début :

Jour🡨 lire ("---")

Mois🡨lire ("---")

Année🡨lire ("---")

Si estValide (jour, mois, année) alors

JourL🡨jour +1

MoisL🡨moi

AnnéeL🡨 année

Si jourL>nombreJour (mois, année) alors

JourL🡨1

MoisL🡨moisL+1

Si moisL>12 alors

MoisL🡨1

Année🡨annéeL +1

FinSi

FinSi

Écrire (jourL+moisL+annéeL)

Sinon

Écrire (date non valide)

FinSi

Fin

Exercice 3 : nombres Premiers

1)

Fonction : comptePremiers (min : entier, max : entier) : entier

Déclaration :

Variables locales :

I, n : entier

Début :

N🡨 0

Pour i allant de min à max faire

N🡨n+1

FinSi

FinPour

Retourner n

Fin

2)

Fonction : premier (rand : entier) : entier

Déclaration :

Variables locales :

I, n : entier

Début :

N🡨1

I🡨2

TantQue n ! = rang Faire

I🡨i+1

Si estPremier (i) Alors

N🡨n+1

FinSi

FinTantQue

Retourner i

Fin

3)

Fonction : rang (premier : entier) : entier

Déclaration :

Variables locales :

I, n : entier

Début :

N🡨0

Pour i allant de 2 à nb Faire

Début :

Retourner comptePremiers(2,nb)

Fin

Si estPremier (i) Alors

N🡨n+1

FinSi

FinPour

Retourner n

Fin

4)

Fonction : petitDiviseur (n : entier) : entier

Déclaration :

Variables locales :

I : entier

Début :

I🡨2

TantQue n % i ! = 0 Faire

I🡨i+1

FinTantQue

Retourner i

Fin

5)

Fonction : estPremier (n : entier) : entier

Déclaration :

Variables locales : Ø

Début :

Retourner petitDiviseur (n) == n

Fin

Exercice 4 :

1)

Fonction : somme (n : entier) : entier

Déclaration :

Variables locales :

Résultat, i : entier

Début :

Résultat🡨0

Pour i allant de 1 à n Faire

Résultat 🡨 résultat + u(i)

FinPour

Retourner résultat

Fin

Fonction : product (n : entier) : entier

Déclaration

Variables locales :

Résultat, i : entier

Début :

Résultat 🡨 1

Pour i allant de 0 à n Faire

Résultat 🡨 résultat \* a (i)

FinPour

Retourner résultat

Fin

Fonction : marc (n : entier) : entier

Déclaration :

Variables locales :

Résultat, i : entier

Début :

Résultat 🡨1

Pour i allant de 1 à n Faire

Si u(i) > résultat Alors

Résultat 🡨 u(i)

FinSi

FinPour

Retourner résultat

Fin

**COURS 4 : les tableaux**

Plan de la séance :

* Tableaux
  + Définition
  + Accès
  + Algorithmes
* Les tableaux en Java
* Algorithme et tableaux
  + Recherche
  + Tri

Problème : trouver le maximum entre 4 (en général n) nombres

Algorithme : Max4

Déclarations :

Variables locales :

A, b, c, d, max : entier

Début :

A ← lire ()

//idem pour b, c, d

Max ← a

Si b > max Alors

Max ← b

FinSi

Si c > max Alors

Max ← c

FinSi

Si d > max Alors

Max ← d

FinSi

Écrire (max)

Fin

Remarques :

* Dans cet algorithme
  + On recommence plusieurs fois le même test
  + Mais on ne peut pas utiliser de boucle
  + Car les tests portent sur des variables différentes
* Solution
  + On crée une structure commune pour l'ensemble des variables et on numérote chaque composant
  + Cette structure est appelée un *tableau*. Tout élément de la structure est une variable qui est accessible par son *indice*

**I/ Les tableaux**

Tableau :

* Un tableau (ou vecteur) est une suite d'éléments de même type, stockés dans des blocs de mémoire contigus, et désignés sous un nom commun
* T : tableau d'entiers initialisé à 1,3, 2, 5, 2
* Les cases d'un tableau sont numérotées
  + A partir de 0
  + Jusqu'à n-1 (n désignant la taille du tableau)
* On peut accéder dans un ordre quelconque à n'importe quel élément : accès direct
* T[1] a pour valeur 3
* T[4] a pour valeur 2
* T[-4] n'est pas défini, de même que t[5]
* T[i] donne l'accès (la valeur) au (i+1)-ième élément du tableau
* Il est possible d'affecter une valeur à t[i]
  + T[3]=0

Déclaration, allocation et affectation :

* Déclaration d'un tableau
  + T : tableau de <type primitif>
* On doit toujours préciser la taille d'un tableau et l'allouer avant de lire ou écrire dedans
  + T🡨 allouer (10)
* La fonction *allouer* réserve le nombre de cases passé en paramètre et retourne une référence sur le tableau ainsi créé
  + Si t référençait un autre tableau avant l'appel, cette référence est perdue
* Important : l'affectation d'une variable tableau à une autre affecte la référence du tableau,
* Elle ne copie pas le tableau
  + Si on veut copier le contenu d'un tableau dans un autre, il faut copier les éléments un par un par une boucle

Retour sur l'algorithme max4

Exemple 1 :

Exemple 2 :

Parcours d'un tableau :

* La méthode de parcours la plus simple consiste à parcourir le tableau dans l'ordre croissant de ses indices : accès séquentiel
  + Boucle Pour allant de 0 à taille(t)-1
* Exemple 3 : procédure afficherTab
* On peut parcourir un tableau pour récupérer une valeur ou un emplacement spécifique
* Exemple 4 : écrire une fonction qui retourne l'indice du tableau où se trouve l'élément maximum

Recherche séquentielle dans un tableau

* Principe : on parcourt le tableau jusqu'à ce que l'une des deux propriétés soit vraie :
  + L'élément courant est la valeur recherchée
  + Le tableau a été entièrement parcouru
* Si t= {9, 2, 5, 7, 3}
  + Recherche (t, 7) vaudra 3
  + Recherche (t, 9) vaudra 0
  + Recherche (t, 8) vaudra -1 (convention)
* On recherche la valeur v dans le tableau
* Si v est dans le tableau
  + Résultat : l'indice où est stocké v
* Si v ne figure pas dans le tableau
  + Résultat =-1 (convention)
* Exemple 5
  + Fonction rechercher qui effectue la recherche séquentielle dans un tableau passé en paramètre

Passage de tableau en paramètre et en retour de fonction :

**Passage en paramètre**

* Pas de recopie du tableau
* Seule la référence du tableau est passée en paramètre
  + Les modifications d'un tableau à l'intérieur d'une fonction/procédure sont permanents
* Exemple 6 : échange de 2 éléments d'un tableau
  + Etat mémoire

**Retour**

* Seule la référence du tableau est retournée au programme appelant
* Le tableau sera généralement déclaré, alloué et initialisé dans la fonction
* Si un tableau est passé en paramètre et modifié, on n'a pas à le retourner
* Exemple 7 : création d'un tableau inversé

**II/ les tableaux en java**

**III/ algorithmes sur les tableaux**