Développement de Sous-programmes utiles en analyse et algorithmique

1. Une procédure qui permet le remplissage d'un tableau T de n entiers **impairs ou premiers et strictement inférieur à 100**

Analyse de la procédure remplissage	Algorithme de la procédure remplissage
DEF PROC remplissage(var t : tab ; n:entier) Resultat = t T = [ ] **Pour i de 1 à n faire** répéter t[i] =donnée (" T[", i , "] = ") jusqu'à **((t[i] mod 2 <>0) ou (FN premier(t[i])) et (t[i] < 100)** **finPour** fin remplissage	0) DEF PROC remplissage(var t : tab ; n:entier) 1) T = [ ] **Pour i de 1 à n faire** répéter Ecrire ("T[", i ,"] = ") Lire(t[i]) jusqu'à **((t[i] mod 2 <>0) ou (FN premier(t[i])) et (t[i] < 100)** **finPour** 2) fin remplissage
Tableau de déclaration des objets locaux : Objet / Nature/Type / Rôle **i** / var/entier / compteur **premier** / Fonction/booléen / Fonction qui décide la primalité d'un entier donné	**Appel de la procédure dans une analyse principale :** Proc remplissage(t,n) Avec t représente une variable globale de type tab passée par variable **[T représente le résultat de la procédure]** et n une variable globale de type entier passée par valeur dit aussi paramètres effectifs

❖ **Analyse et algorithme de la fonction premier :**

Analyse de la fonction premier	Algorithme de la fonction premier
DEF FN premier (x : entier) : booléen Resultat = premier ← nbd = 2 **nbd = [nbd ← 1]** **pour i de 2 à x faire** **[ ] si x mod i = 0 alors nbd ←nbd + 1 finSi** **finPour** fin premier **Notez bien que :** Resultat = premier ← nbd = 2 remplace la structure Resultat = premier Premier = [ ] si nbd = 2 alors premier ← vrai Sinon premier ← faux finSi	0) DEF FN premier (x : entier) : booléen 1) **[nbd ← 1]** **pour i de 2 à x faire** **[ ] si x mod i = 0 alors nbd ←nbd + 1 finSi** **finPour** 2) premier ← nbd = 2 3) fin premier **Notez bien que :** 2) premier ← nbd = 2 remplace la structure 2) [ ] si nbd = 2 alors premier ← vrai Sinon premier ← faux finSi
Tableau de déclaration des objets locaux : Objet / Nature/Type / Rôle **i** / var/entier / Compteur **nbd** / Var/entier / Nbre de diviseurs	**Appel de la fonction dans une analyse principale ou un autre module(fonction ou procédure):** B ← FN premier(n) Avec n représente une variable de type entier aussi dit un paramètre global effectif et B une variable globale de type booléen

2. Une procédure qui permet le remplissage d'un tableau T de n chaines de **6 caractères** contenant que des **lettres majuscules**

Analyse de la procédure remplissage	Algorithme de la procédure remplissage
DEF PROC remplissage(var t : tab ; n:entier) Resultat = t T = [ ] **Pour i de 1 à n faire** répéter	0) DEF PROC remplissage(var t : tab ; n:entier) 1) T = [ ] **Pour i de 1 à n faire** répéter Ecrire ("T[", i ,"] = ")

t[i] =donnée (" T[", i , "] = ") jusqu'à **(FN majuscule(t[i])) ET (long(t[i]) = 6)** **finPour** fin remplissage	Lire(t[i]) jusqu'à **(FN majuscule(t[i])) ET (long(t[i]) = 6)** **finPour** 2) fin remplissage
Tableau de déclaration des objets locaux : Objet / Nature/Type / Rôle **i** / var/entier / compteur **majuscule** / Fonction/booléen / Fonction qui décide que les caractères d'une chaine donnée sont tous majuscules	**Appel de la procédure dans une analyse principale :** Proc remplissage(t,n) Avec t représente une variable globale de type tab passée par variable **[T représente le résultat de la procédure]** et n une variable globale de type entier passée par valeur dit aussi paramètres effectifs

❖ Analyse et algorithme de la fonction premier :

Analyse de la fonction premier	Algorithme de la fonction premier
DEF FN majuscule (ch : chaine) : booléen Resultat = majuscule ← b **b = [ ] si i > long(ch) alors b ← vrai** **[i ←1] répéter** **si ch[i] dans ["A".."Z"] alors** **i ←i + 1** **sinon b ←faux** **finSi** **jusqu'à (b = faux) ou (i >long(ch))** fin majuscule **Notez bien la condition b = faux peut être remplacé par non (b)**	0)DEF FN majuscule (ch : chaine) : booléen **1) [i ←1] répéter** **si ch[i] dans ["A".."Z"] alors** **i ←i + 1** **sinon b ←faux** **finSi** **jusqu'à (b = faux) ou (i >long(ch))** 2) **si i > long(ch) alors b ← vrai** 3) majuscule ← b 4)fin majuscule **Notez bien la condition b = faux peut être remplacé par non (b)**
Tableau de déclaration des objets locaux : Objet / Nature/Type / Rôle **i** / var/entier / Compteur **b** / Var/booléen / Décision	**Appel de la fonction dans une analyse principale ou un autre module(fonction ou procédure):** B ← FN majuscule(ch) Avec ch représente une variable de type chaine aussi dit un paramètre global effectif et B une variable globale de type booléen

3. Une procédure qui permet le remplissage d'un tableau T de n chaines d'**au moins 10 caractères** tel que **le premier caractère de chaque élément du tableau n'est pas une voyelle**

Analyse de la procédure remplissage	Algorithme de la procédure remplissage
DEF PROC remplissage(var t : tab ; n:entier) Resultat = t T = [ ] **Pour i de 1 à n faire** répéter t[i] =donnée (" T[", i , "] = ") jusqu'à **(non(majus(t[i][1]) dans ["A","E","O","U","I","Y"])) ET (long(t[i]) >= 10)** **finPour** fin remplissage	0) DEF PROC remplissage(var t : tab ; n:entier) 1) T = [ ] **Pour i de 1 à n faire** répéter Ecrire ("T[", i ,"] = ") Lire(t[i]) jusqu'à **(non(majus(t[i][1]) dans ["A","E","O","U","I","Y"])) ET (long(t[i]) >= 10)** **finPour** 2) fin remplissage
Tableau de déclaration des objets locaux : Objet / Nature/Type / Rôle **i** / var/entier / compteur	**Appel de la procédure dans une analyse principale :** Proc remplissage(t,n) Avec t représente une variable globale de type tab passée par variable **[T représente le résultat de la procédure]** et n une variable globale de type entier passée par valeur dit aussi paramètres effectifs

4. Une procédure qui permet le remplissage d'un tableau T de n chaines de caractères **autres que des chiffres**

Analyse de la procédure remplissage	Algorithme de la procédure remplissage
DEF PROC remplissage(var t : tab ; n:entier) Resultat = t T = [ ] **Pour i de 1 à n faire** répéter t[i] =donnée (" T[", i , "] = ") jusqu'à **non (FN chiffre(t[i]))** **finPour** fin remplissage	0) DEF PROC remplissage(var t : tab ; n:entier) 1) T = [ ] **Pour i de 1 à n faire** répéter Ecrire ("T[", i ,"] = ") Lire(t[i]) jusqu'à **non (FN chiffre(t[i]))** **finPour** 2) fin remplissage
Tableau de déclaration des objets locaux : Objet | Nature/Type | Rôle **i** | var/entier | compteur **chiffre** | Fonction/booléen | Fonction qui décide que les caractères d'une chaine donnée sont tous des chiffres	**Appel de la procédure dans une analyse principale :** Proc remplissage(t,n) Avec t représente une variable globale de type tab passée par variable **[T représente le résultat de la procédure]** et n une variable globale de type entier passée par valeur dit aussi paramètres effectifs

❖ Analyse et algorithme de la fonction chiffre :

Analyse de la fonction Chiffre	Algorithme de la fonction Chiffre
DEF FN chiffre (ch :chaine) : booléen Resultat = Chiffres **Chiffre = [ ]** **Si i<=long(ch) alors** **chiffres ←faux** **sinon chiffres ←vrai** **finSi** **[i ← 0]** **répéter** **i ←i+1** **jusqu'à (non (ch[i] dans ["0".."9) ou (i>long(ch))** fin chiffre	0) DEF FN chiffre (ch :chaine) : booléen 1) **[i ← 0]** **répéter** **i ←i+1** **jusqu'à (non (ch[i] dans ["0".."9) ou (i>long(ch))** 2) **[ ]** **Si i<=long(ch) alors chiffre ←faux** **sinon chiffre ←vrai** **finSi** 3)fin chiffres
Tableau de déclaration des objets locaux : Objet | Nature/Type | Rôle **i** | var/entier | Compteur	**Appel de la fonction dans une analyse principale ou un autre module(fonction ou procédure):** B ← FN chiffre(ch) Avec ch représente une variable de type chaine aussi dit un paramètre global effectif et B une variable globale de type booléen

5. Une procédure qui permet **l'affichage d'un tableau composé par n entiers**

Analyse de la procédure affichage	Algorithme de la procédure affichage
DEF PROC affichage(t : tab ; n : entier) Resultat = RetourLigne [ ] **Pour i de 1 à n faire** Ecrire(t[i], " | ") **finPour** fin affichage **Notez bien : RetourLigne est une instruction qui permet le retour à la ligne son équivalent en pascal writeln;**	0) DEF PROC affichage(t : tab ; n : entier) 1) **Pour i de 1 à n faire** Ecrire(t[i], " | ") **finPour** 2) RetourLigne 3) fin remplissage **Notez bien : RetourLigne est une instruction qui permet le retour à la ligne son équivalent en pascal writeln;**
Tableau de déclaration des objets locaux : Objet | Nature/Type | Rôle **i** | var/entier | compteur	**Appel de la procédure dans une analyse principale :** Proc affichage(t,n) Avec t représente une variable globale de type tab passée par valeur et n une variable globale de type entier passée par valeur dit aussi paramètres effectifs

6. Une procédure qui permet **l'affichage** d'un tableau **dans l'ordre inverse** composé **par n entiers**

Analyse de la procédure affichage	Algorithme de la procédure affichage
DEF PROC affichage(t : tab ; n : entier) Resultat = RetourLigne [ ] **Pour i de n à 1 (pas = - 1) faire** Ecrire(t[i], " | ") **finPour** fin affichage	0) DEF PROC affichage(t : tab ; n : entier) 1) **Pour i de n à 1 (pas = - 1) faire** Ecrire(t[i], " | ") **finPour** 2) RetourLigne 3) fin remplissage
Tableau de déclaration des objets locaux : Objet | Nature/Type | Rôle **i** | var/entier | compteur	**Appel de la procédure dans une analyse principale :** Proc affichage(t,n) Avec t représente une variable globale de type tab passée par valeur et n une variable globale de type entier passée par valeur dit aussi paramètres effectifs

7. Une fonction qui permet la vérification qu'un entier est parfait (un entier parfait **est égal** à la **somme de ses diviseurs propres** par exemple : 6 = 1 + 2 +3)

Analyse de la fonction parfait	Algorithme de la fonction parfait
DEF FN parfait(x : entier) : booléen Resultat = parfait ← s = x **[s ← 0] pour i de 1 à x div 2 faire** **[ ] si x mod i = 0 alors s ←s + i** **finPour** fin parfait **Notez bien : l'instruction** parfait ← s = x peut être remplacé par **Resultat = parfait** **Parfait = [ ] si s = x alors parfait ←vrai** **sinon parfait ← faux** **finSi**	0) DEF FN parfait(x : entier) : booléen 1) **[s ← 0] pour i de 1 à x div 2 faire** **[ ] si x mod i = 0 alors s ←s + i** **finPour** 2) parfait ← s = x 3)fin parfait **Notez bien : l'instruction 2)** parfait ← s = x peut être remplacé par **2)si s = x alors parfait ←vrai** **sinon parfait ← faux** **finSi**
Tableau de déclaration des objets locaux : Objet | Nature/Type | Rôle **i** | var/entier | Compteur **s** | Var/entier | Somme des diviseurs propres	**Appel de la fonction dans une analyse principale ou un autre module(fonction ou procédure):** B ← FN parfait(n) Avec n représente une variable de type entier aussi dit un paramètre global effectif et B une variable globale de type booléen

8. Une fonction qui permet la vérification qu'une chaine est un palindrome (Un palindrome est une chaine de caractères qui peut être lue du **gauche à droite** et du **droite à gauche** est **elle s'agit de la même chaine**) par exemple "ACBCA" est un palindrome par contre "ACBA" ne l'est pas

Analyse de la fonction parfait	Algorithme de la fonction parfait
DEF FN palindrome(sh : chaine) : booléen Resultat = palindrome ← i > j **[i ← 1 , j ← long(sh)]** **Tant que** (i <= j) et (**sh[i] = sh[j]**) **faire** **i ←i + 1** **j ←j - 1** **finTantQue** fin palindrome **Notez bien : l'instruction** palindrome ← i>j peut être remplacé par **Resultat = palindrome** **Parfait = [ ] si s = x alors parfait ←vrai** **sinon parfait ← faux finSi**	0) DEF FN palindrome(sh : chaine) : booléen 1) **[i ← 1 , j ← long(sh)]** **Tant que** (i <= j) et (**sh[i] = sh[j]**) **faire** **i ←i + 1** **j ←j - 1** **finTantQue** 2) palindrome ← i>j 3)fin palindrome **Notez bien : l'instruction 2)** palindrome ← i>j peut être remplacé par **2)si i>j alors palindrome ←vrai** **sinon palindrome ← faux** **finSi**

Langage de programmation : Pascal

Tableau de déclaration des objets locaux :

Objet	Nature/Type	Rôle
i	var/entier	Compteur de gauche à droite
j	Var/entier	Compteur de droite à gauche

Appel de la fonction dans une analyse principale ou un autre module(fonction ou procédure):

B ← FN palindrome(ch)

Avec ch représente une variable de type chaine aussi dit un paramètre global effectif et B une variable globale de type booléen

9. Une fonction qui permet le calcul le **nombre d'occurrences** d'un **caractère x donné** (Le nombre d'occurrences est le nombre de répétitions) **dans une chaine ch**

Analyse de la fonction occurrence

```
DEF FN occurrence(ch : chaine; x:caractère) : booléen
  Resultat = occurrence ← nbo
  nbo = [ nbo ← 0 ]
    Pour i de 1 à long(ch) faire
              [ ] si ch[i] = x alors nbo ←nbo+ 1 finSi
    finPour
  fin occurrence
```

Algorithme de la fonction occurrence

```
0) DEF FN occurrence(ch : chaine; x:caractère) : booléen
  1) [ nbo ← 0 ]
    Pour i de 1 à long(ch) faire
              [ ] si ch[i] = x alors nbo ←nbo+ 1 finSi
    finPour
2) occurrence ← nbo
3)fin occurrence
```

Tableau de déclaration des objets locaux :

Objet	Nature/Type	Rôle
i	var/entier	compteur
nbo	Var/entier	Nbre d'occurrence x dans ch

Appel de la fonction dans une analyse principale ou un autre module(fonction ou procédure):

n ← FN occurrence(ch, c)

Avec ch représente une variable de type chaine et c une variable de type caractère passés les deux par valeur aussi dit des paramètres globaux effectifs et n une variable globale de type entier

10. Une fonction qui permet le calcul du factorielle d'un entier n donné(factorielle de n! = n * n-1*n-2* . . . * 1)

Analyse de la fonction factorielle

```
DEF FN factorielle(n : entier) : entier
  Resultat = factorielle ← f
  f = [ f ← 1 ] Pour i de 1 à n faire
                        f ←f * i
          finPour
  fin factorielle
```

Algorithme de la fonction factorielle

```
0) DEF FN factorielle(n : entier) : entier
  1 [ f ← 1 ] Pour i de 1 à n faire
                        f ←f * i
          finPour
2) factorielle ← f
3)fin factorielle
```

Tableau de déclaration des objets locaux :

Objet	Nature/Type	Rôle
i	var/entier	compteur
f	Var/entier	f = n!

Appel de la fonction dans une analyse principale ou un autre module(fonction ou procédure):

n ← FN factorielle(x)

Avec x représente une variable de type passé par valeur aussi dit paramètre global effectif et n une variable globale de type entier