***1er BTS SIO ALGORITHMIQUE***

***CHAPPITRE V CHAINES DE CARACTERES , TABLEAUX***

1. ***Chaines de caractères***

***Définition***

* *Dans un programme informatique, les chaînes de caractères servent à stocker les informations non numériques comme par exemple une liste de nom de personne ou des adresses. Une chaîne de caractères est une suite ordonnée de caractères.*

*Un caractère peut-être :*

*• une lettre (caractère alphabétique – la distinction est faite entre les lettres minuscules et les lettres majuscules),*

*• un chiffre (caractère numérique) ,*

*• un signe de ponctuation,*

*• ou tout autre symbole (comme l'étoile, le tiret, le "slash"...).*

* *En algorithmique, une chaîne de caractères est considérée comme une variable.*
* *En Python, les chaînes de caractères sont classées dans les séquences tout comme les listes et les tuples (les tuples et les chaînes sont des séquences immuables, c.à.d. qu'elles ne peuvent pas être modifiées).*

***Exemples*** *:*

*• La chaîne de caractères "Baobab" est constituée des caractères "B", "a", "o", "b", "a" et "b".*

*• La chaîne de caractères "123" n'est constituée que de chiffres, mais en aucun cas il ne faut la confondre avec la grandeur numérique 123.*

*• La chaîne de caractères "" représente une chaîne de caractères vide.*

*• La chaîne de caractères " " est une chaîne de caractères ne contenant qu'un seul caractère qui est ici le caractère espace (à ne pas confondre avec la chaîne de caractères vide).*

***Rang d’un caractère***

*Les chaînes sont constituées de caractères. Chaque caractère (qui n'est autre qu'un symbole) se trouve à une position donnée dans la chaîne (on parle alors de rang). Le rang d'un caractère désigne son classement la chaîne.*

*En algorithmique, le premier caractère se trouve au rang 1, le deuxième caractère se trouve au rang 2 et ainsi de suite...*

*En langage Python, le premier caractère d'une chaîne se trouve au rang 0, le deuxième caractère se trouve au rang 1 et ainsi de suite...*

***Exemple :*** *Chaîne "Baobab"*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Rangs (python)* | *0* | *1* | *2* | *3* | *4* | *5* |
| *caractères* | *B* | *a* | *o* | *b* | *a* | *b* |
| *Rangs (algo)* | *1* | *2* | *3* | *4* | *5* | *6* |

***Comparaison de deux chaines de caractères***

***code ASCII***

*Dans un système informatique, à chaque caractère est associée une valeur numérique : son code ASCII (American Standard Code for Information Interchange). L'ensemble des codes est recensé dans une table nommée "table des codes ASCII". Quand on stocke un caractère en mémoire (dans une variable), on mémorise en réalité son code ASCII. Un code ASCII est codé sur un octet (huit bits). La table des codes ASCII est donnée à la fin de ce document.*

***Comparaison de caractères***

*"B" > "A" car le code ASCII du caractère "B" (66 en base 10) est supérieur au code ASCII du caractère "A" (65 en base 10). Pour comparer deux chaînes de caractères, on compare les caractères de même rang dans les deux chaînes en commençant par le premier caractère de chaque chaîne (le premier caractère de la première chaîne est comparé au premier caractère de la seconde chaîne, le deuxième caractère de la première chaîne est comparé au deuxième caractère de la seconde chaîne, et ainsi de suite…).*

***Exemples : Comparaison de deux chaînes***

*• "baobab"> "banania" car le code ASCII de "o" est supérieur au code ASCII de "n" (la comparaison ne peut pas se faire sur les deux premiers caractères car ils sont identiques).*

*• "1999" > "1998" car le code ASCII de "9" est supérieur au code ASCII de "8" (la comparaison ne peut pas se faire sur les trois premiers caractères car ils sont identiques). Attention, ici ce ne sont pas des valeurs numériques qui sont comparées, mais bien des caractères.*

*• "333" > "1230" car le code ASCII de "3" est supérieur au code ASCII de "1".*

*• "333"< "3330" car la seconde chaîne a une longueur supérieure à celle de la première (la comparaison ne peut pas se faire sur les trois premiers caractères car ils sont identiques).*

*• "Baobab"  < "baobab" car le code ASCII de "b" est supérieur au code ASCII de "B".*

***Exercice d'application*** *:*

*Compléter le tableau ci-dessous en indiquant la relation d'ordre () ainsi que le nombre de caractères concernés par la comparaison*.

|  |  |  |  |
| --- | --- | --- | --- |
| "python" | > | "Python" | Le code ASCII de "p" est supérieur au code ASCII de "P". |
| "Bonsoir" | > | "Bonjour" | Le code ASCII de « s » est supérieur au code ASCII de « j ». |
| "Hola" | < | "hola" | Le code ASCII de « h » est supérieur au code ASCII de « H ». |
| "Ohé" | > | "Eho" | Le code ASCII de « O » est supérieur au code ASCII de « E ». |
| "BTS1SIO" | < | "BTS2IG" | Le code ASCII de « 2 » est supérieur au code ASCII de « 1 ». |
| "28/02/17" | < | "28/02/18" | Le code ASCII de « 8 » est supérieur au code ASCII de « 7 ». |
| "C & Java" | < | "C | Java" | Le code ASCII de « | » est supérieur au code ASCII de « & ». |
| "> 15" | > | "< 15" | Le code ASCII de « > » est supérieur au code ASCII de « < ». |
| "toi" | = | "toi" |  |
| "5170" | > | "517" | Il y a 0 en plus. |
| "318" | > | "1273" | Le code ASCII de « 3 » est supérieur au code ASCII de « 1 ». |

***Concaténation de chaînes de caractères***

*Le but de la concaténation est de créer des chaînes de caractères en juxtaposant deux (ou plus) chaînes de caractères.*

*En algorithmique, l'opérateur de concaténation est représenté par +*

*L'opération de concaténation peut être appliquée à une même variable (tout comme pour les variables dites "de cumul" ou les variables "compteurs").*

***Exemple*** *:* ***Concaténation de deux chaînes***

*Algo Concaténations*

*Variable :*

*catégorie : Chaîne*

*Début*

*catégorie ← "développeur"*

*catégorie← catégorie + "s"*

*Afficher(catégorie)                 # Affiche : développeurs*

*catégorie ← "des " + catégorie*

*Afficher(catégorie)                 # Affiche : des développeurs*

*Fin*

***Manipulation de chaînes de caractères***

***Copie d'un extrait***

*Algo. SSCHAINE(chaîne,position,nombre):Chaîne*

*Python chaîne[position]*

*chaîne[position\_début:position\_fin\_exclue]*

*Le rôle de la fonction SSCHAINE() est de retourner une "sous-chaîne" (copie d'un extrait de la chaîne passée en premier paramètre) de nombre caractères de la chaîne* ***chaîne*** *à partir du caractère qui se trouve en position* ***position*** *(rappel : le premier caractère d'une chaîne de caractères se trouve en position 1 en algorithmique et en position 0 en langage Python). En Python, une chaîne étant une séquence, l'utilisation des crochets permet d'accéder à un extrait constitué d'un ou plusieurs caractères.*

***Nombre de caractères***

*Algo. LONGUEUR(chaîne):Entier*

*Python len(chaîne)*

*La fonction LONGUEUR() a pour rôle de calculer et de retourner le nombre de caractères présents dans la chaîne de caractères passée en paramètre. En Python, on peut remarquer que la fonction est identique à celle utilisée avec les listes et les tuples. En fait, la fonction len() permet de connaître la longueur (nombre d'éléments) d'une séquence... que ce soit une liste, un tuple ou une chaîne.*

***Recherche d'un caractère ou d'une sous-chaîne***

*Algo RANG(chaîne,sous­chaîne,position):Entier*

*Python chaîne.index(sous­chaîne,position)*

*chaîne.find(sous­chaîne,position)*

*La fonction RANG() recherche et retourne la première occurrence de la sous­chaîne dans la*

*chaîne. La recherche commence à partir de position. Cette fonction retourne la position du*

*premier caractère de la sous­chaîne, ou la valeur 0 si la "sous­chaîne" n'existe pas dans la*

*chaîne (à partir de position). En Python, si aucune occurrence de la sous-chaîne n'est trouvée, la méthode find() retourne la valeur ­1 alors que la méthode index() génère une exception(ValueError). 4.4.*

***Conversion en code ASCII***

*Algo CODE(caractère):Entier*

*Python ord(caractère)*

*La fonction CODE() retourne le code ASCII du caractère passé en paramètre.*

***Conversion en caractère***

*Algo CAR(code\_ascii):Caractère*

*Python chr(code\_ascii)*

*La fonction CAR() retourne le caractère dont le code ASCII est passé en paramètre.*

***Conversion en grandeur numérique***

*Algo CVNOMBRE(chaîne):Entier*

*CVNOMBRE(chaîne):Réel*

*Python int(chaîne)*

*float(chaîne)*

*La fonction CVNOMBRE() a pour rôle de convertir (CV pour ConVersion) la chaîne de caractères passée en paramètre, en grandeur numérique si la suite de caractères représente un nombre, puis de retourner la grandeur numérique.*

*Pour stocker en mémoire la valeur numérique 65534 (0xFFFE), il faut deux octets (puisque avec un seul octet on peut coder une valeur comprise entre 0 et 255 inclus). Par contre, pour stocker la chaîne de caractères "65534", il faut cinq octets en mémoire puisque cette chaîne contient cinq caractères, et qu'il faut un octet par caractère.*

***Exemple*** *:* ***Occupation mémoire***

*Variables :*

*valeur\_chaîne : Chaîne*

*valeur\_entier : Entier*

*Début*

*valeur\_chaîne ← "65534"*

*valeur\_entier ← 65534*

*...*

*Fin*

*Dans la réalité, les nombres entiers (ainsi que les nombres réels) sont tous codés sur le même*

*nombre d'octets. Ce nombre d'octets dépend du système d'exploitation et du microprocesseur*

*(pour les entiers, on parle de "mot machine") et non de la valeur à mémoriser.*

***Conversion en chaîne de caractères***

*Algo CVCHAINE(entier):Chaîne*

*CVCHAINE(réel):Chaîne*

*Python str(entier)*

*str(réel)*

*La fonction CVCHAINE() a pour rôle de convertir en chaîne de caractères une valeur numérique passée en paramètre, puis de retourner la chaîne résultante.*

***Exercice d'application*** *:*

*Que retourne chacune des instructions ci-dessous ? Traduire chaque instruction en langage Python.*

|  |  |  |  |
| --- | --- | --- | --- |
| *Algo* | *Algo* | *Python* | *Python* |
| *LONGUEUR("Agenda")* | *6* | *6* | *len("Agenda")* |
| *SSCHAINE("Almanach",2,3)* | *lma* | *lma* | *Almanach[1:4]* |
| *SSCHAINE("Almanach",5,1)* | *n* | *n* | *Almanach[4:5]* |
| *SSCHAINE("Almanach",5)* | *n* | *n* | *Almanach[4]* |
| *RANG("Almanach","an",1)* | *4* | *3* | *Almanach.index(« an »,0)*  *ou*  *Almanach.find(« an »,0)* |
| *RANG("Almanach","a")* | *4* | *3* | *Almanach.index(« a »)*  *ou*  *Almanach.find(« a »)* |
| *RANG("Almanach","a",4)* | *4* | *3* | *Almanach.index(« a »,3)*  *ou*  *Almanach.find(« a »,3)* |
| *RANG("Almanach","kan",1)* | *erreur* | *erreur* | *Almanach.index(« kan »,0)*  *ou*  *Almanach.find(« kan »,0)* |
| *RANG("Almanach","a",5)* | *6* | *5* | *Almanach.index(« a »,4)*  *ou*  *Almanach.find(« a »,4)* |
| *CODE("Z")* | *90* | *90* | *ord(« Z »)* |
| *CAR(118)* | *v* | *v* | *chr(118)* |

*Pour conclure...*

*En Python, il existe de nombreuses fonctions/méthodes de manipulation des chaînes de caractères comme celles qui n'ont pas été détaillées dans ce document. Certaines sont applicables aux séquences en général et d'autres sont spécifiques aux chaînes. Elles peuvent être classées en plusieurs catégories :*

*• Modification de la casse : lower(), upper(), swapcase(), capitalize(), title()*

*• Nombre d'occurrences : count()*

*• Suppression de caractères : lstrip(), rstrip(), strip()*

*• Alignement : ljust(), rjust(), center()*

*• Découper et reconstituer : split(), join()*

*• Remplacement : replace()*

*• Recherche inversée : rindex(), rfind()*

*• Valeur minimale et valeur maximale : min(), max()*

*• Classe des caractères : isalnum(), isalpha(), isdigit(), islower(), isupper(),*

*isspace(), istitle()*

*Pour vous aider à utiliser au mieux toutes ces fonctions, il existe la documentation Python que vous pouvez consulter sur Internet, la commande pydoc ainsi que l'interpréteur en mode interactif.****II. TABLEAUX***

*Pour stocker les nombres en mémoire, on utilise un type de variable qui peut mémoriser plusieurs valeurs : les tableaux.*

1. ***Tableau à une dimension***

*1° /****Définition***

*Un tableau est une structure qui permet de stocker sous le même nom un ensemble de données de même type.*

*Un tableau possède une taille fixe connue dès le départ de l’algorithme, cela correspond au nombre de cases du tableau, appelés aussi nombre d’éléments du tableau.*

*2°/* ***Remarques***

*Un tableau peut ne pas être entièrement rempli mais ne pourra jamais contenir plus d’élément que sa taille déclarée.*

*On ne peut pas accéder à l’ensemble d’un tableau, on accède à une case du tableau à la fois. Le nom du tableau ne sera donc jamais utilisé seul, mais suivi par le numéro de la case concernée placé entre crochets. Ce numéro de case est appelé indice.*

*L’indice est une variable qui sert à se déplacer de case en case dans le tableau, il peut commencer à 0 ou à 1(selon le langage de programmation en python : il commence à 0). En algorithmique i commence à 1.*

*On dit qu’un tableau est à une dimension lorsqu’un seul indice suffit à repérer un élément du tableau.*

*Ne pas confondre l’indice (le numéro de la case) et la valeur contenue dans cette case.*

*3°/* ***Déclaration d’un tableau***

*Un tableau doit être déclaré en début de programme, dans la partie déclaration des variables, et la déclaration doit comporter un nom, une taille et un type.*

***Exemple*** 

*nomTab : tableau[1..max] de type :entier, réel, caractères ou tableau: nomTab[1..max] de type :entier, réel, caractères*

*un nom :nomTab, une taille : de 1 à max éléments dans le tableau, un type : type tab(entier, réel, caractères) : c’est le type de tous les éléments stockés dans le tableau.*

***Remarque***

*L’indice « le curseur » qui va servir à se déplacer dans le tableau pour indiquer le numéro de la case est un entier qui doit être déclaré indépendamment du tableau.*

***4°/Affichage et remplissage d’un tableau***

*Pour afficher le nième élément du tableau, on écrit afficher NOM DU TABLEAU[n]*

*Pour donner une valeur à chaque case du tableau, on peut procéder de deux manières :*

*Par affectation : nom du tableau[n]**valeur*

*ou par saisie : saisir nom du tableau[n]*

*i**n*

*nom du tableau[i]**valeur*

***Exemple***

*Tablenotes[1..10] est un tableau de 10 notes.*

*Pour afficher la 4ième note on écrit afficher Tablenotes[4]*

***PYTHON****:*

*En python les numéros de cases d’un tableau commencent à 0*

*En python pour afficher le contenu de la nième case d’un tableau Tab, on écrit :*

*print(Tab[n-1]).*

*Pour donner une valeur à la nième case du tableau Tab, on écrit :*

*Tab[n-1]= input()*

***Application 1***

*L’algorithme suivant détermine le minimum de dix nombres saisis*

*Variables : nbre, mini : réels*

*i: entier*

*Début*

*Afficher « veuillez saisir un nombre »*

*Saisir nbre*

*mini* *nbre*

*pour i de 2 à 10 faire*

*saisir nbre*

*si nbre<mini faire*

*mini**nbre*

*fin si*

*Fin pour*

*Afficher « le minimum des dix nombres est : »+mini*

*Fin*

1. *Transformer cet algorithme en utilisant les tableaux pour qu’il mémorise les dix valeurs et affiche leur maximum.*

Variables

nbre maxi: reel

i : entier

tableau : Tab[1...10]: reel

Début

Afficher « saisir un nombre »

Saisir Tab[1]

maxi ← Tab[1]

Pour i de 2 à 10 faire

saisir Tab[i]

Si maxi < Tab[i]

maxi ← Tab[i]

Fin si

Fin pour

Afficher « le maximum est : » , maxi

Fin

1. *Traduire l’algorithme précédent en python et l’exécuter pour le tableau suivant :*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *6* | *15* | *-2* | *17* | *4* | *12* | *5* | *4* | *10* | *1* |

***Application 2***

1. *Que produit l’algorithme suivant ?*

*Cette algorithme demande la saisie de n, remplit et affiche les cases d’n tableau de n cases avec les carrés des n premiers entiers non nuls.*

*Variables*

*Tableau Nb[1..n] Entier*

*i ,n:Entier  
Début*

*saisir n  
Pour i ← 1 à n  
  Nb[i]← i \* i  
Fin pour  
Pour i ← 1 à n  
  Afficher Nb[i]  
Fin pour****Fin***

1. *Peut-on simplifier cet algorithme avec le même résultat ?*

*Oui : remplacer pour i de 1 à n afficher Nb[i] par afficher Nb*

1. *Traduire l’algorithme précédent en python et l’exécuter pour n=10*

Voir dans le dossier python

***Application 3***

1. *Que produit l’algorithme suivant ?*

***Variables***

***Tableau****: N[1..6]****:******Entier***

*i, k****:******Entier******Début*** *N[1] ← 1****Pour*** *k ← 2 à 6 faire :  
  N[k] ← N[k-1] + 2  
Fin pour****Pour*** *i ← 1 à 6 faire : A partir d’ici ! (question 2)* ***Afficher*** *N[i]  
Fin pour****Fin***

1. *Peut-on simplifier cet algorithme avec le même résultat ?*

*Afficher N : à la place des trois dernière lignes*

1. *Traduire l’algorithme précédent en python et l’exécuter.*

*Voir dans le dossier python*

1. ***Tableaux à deux dimensions***
2. *Définition*

*Un tableau à deux dimensions contient plusieurs lignes et plusieurs colonnes. Deux indices seront nécessaires pour utiliser un tel tableau. Un indice lignes et un indice colonnes*

1. *Déclaration d'un tableau*

***NomTab : Tableau [ (1,l)(1 , c )] :Type***

*l : indique le nombre de lignes*

*c : indique le nombre de colonnes*

*En python:NomTab=[][]*

***Application 1***

*Ecrire un algorithme qui permet d'initialiser le contenu d'un tableau à deux dimensions (5 lignes et 6 colonnes) à 0.*

*Variables*

*l, c : entier*

*Tableau : Tab(1..5][1..6] : entier*

*Début*

*pour l de 1 à 5 faire*

*pour c de 1 à 6 faire*

##### Tab [l][c] ← 0

Fin pour

Fin pour

Fin

***En python :*** Tab=[[0 for l in range(5)]for c in range(6)]

pour le mettre au format tableau : for l in range(5) :

print(Tab[l])

*EXERCICE :*

1. *Ecrire un algorithme qui demande la saisie des valeurs de A et affiche A = ;*

*Variables*

*k,l, c : entier*

*Tableau : Tab(1..5][1..6] : entier*

*Début*

*k← 0*

*pour l de 1 à 4 faire*

*pour c de 1 à 4 faire*

##### Tab [l][c] ← k+1

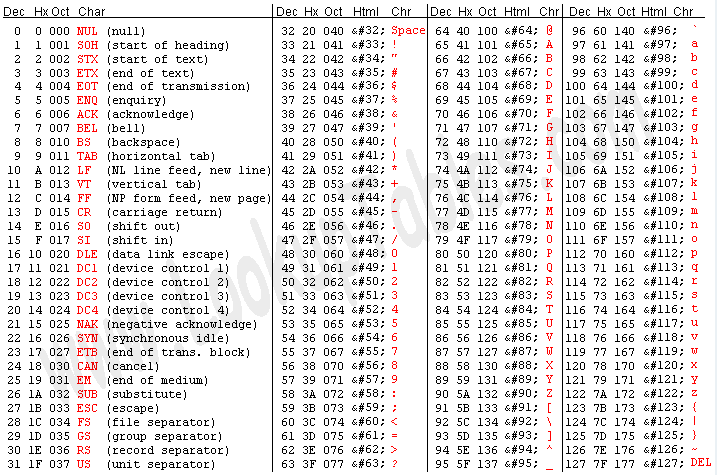
K← k+1

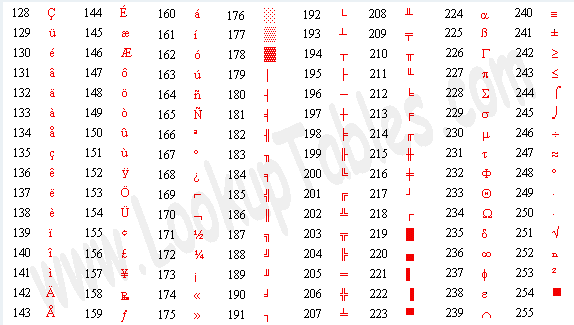
Fin pour

Fin pour

Fin

***TABLE ASCII***

**

**

BREVET DE TECHNICIEN SUPÉRIEUR

SERVICES INFORMATIQUES AUX ORGANISATIONS

**EPREUVE PONCTUELLE**

**ALGORITHMIQUE APPLIQUEE E22 SESSION 2015**