Ajournement

15-09-2003

1. **(8p.)** Développez la fonction `TOUCHE_COULE` en **TurboPascal**. C'est une fonction d'un jeu (en allemand : Schiffeversenken) qui vérifie si un bateau a été touché ou non. La fonction retourne donc '`true`' si le tir a touché un bateau et '`false`' le cas échéant.
   La structure de données du jeu est un tableau `TAB` à deux dimensions du type `tJeu`.

```
type tJeu = array[1..16,1..16] of boolean ;
```

   Un bateau est représenté par des valeurs logiques '`true`' – l'océan est constitué de valeurs logiques '`false`'.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
2	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
3	F	T	F	F	F	F	F	F	F	F	F	T	T	T	T	T
4	F	T	F	F	F	F	F	F	F	F	F	F	F	F	F	F
5	F	T	F	F	F	F	F	F	T	F	F	F	F	F	F	F
6	F	F	F	F	F	F	F	F	T	F	F	F	F	F	F	F
7	F	F	F	F	F	F	F	F	T	F	F	F	F	F	F	F
8	F	F	T	T	T	T	F	F	T	F	F	F	F	F	F	F
9	F	F	F	F	F	F	F	F	T	F	F	F	F	F	F	F
10	F	F	F	F	F	F	F	F	T	F	F	T	T	F	F	F
11	F	F	F	F	F	F	F	F	T	F	F	F	F	F	F	F
12	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
13	F	F	F	T	T	F	F	F	F	F	F	F	F	F	F	T
14	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	T
15	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	T
16	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	T

(`T` = true `F` = false)

   Le tableau ainsi que les coordonnées du tir sont transmis à la fonction. On suppose que les coordonnées du tir sont valables et ne dépassent pas les limites du tableau.
   Attention : Programmez cette fonction avec le moins de lignes de codes nécessaires !

2. **(10p.)** Développez le **structogramme** du sous-programme `INSNOMBRE` qui insert un `NOMBRE` dans un tableau `TAB` du type `tTab`.

```
type tTab = array [1..MAXDIM] of integer.
```

   Le tableau est trié en ordre décroissant avant l'insertion - il doit rester trié en décroissant (après insertion de `NOMBRE` -> voir exemples ci-dessous).
   Pour simplifier l'algorithme, on suppose que la longueur maximale du tableau n'est pas encore atteinte !

Ajournement

15-09-2003

NOMBRE, TAB et la longueur effective N sont transmis au sous-programme.
INSNOMBRE retourne la position à laquelle NOMBRE a été inséré.

Exemple1 :
```
avant : TAB [8,7,6,4,3,2]   NOMBRE = 5
après : TAB [8,7,6,5,4,3,2]
```

Exemple2 :
```
avant : TAB [8,7,6,5,4,3,2] NOMBRE = 5
après : TAB [8,7,6,5,5,4,3,2]
```

3. **(10p.)** Développez la fonction ALPHAPOS sous forme de **structogramme**. La fonction retourne la position d'un caractère dans l'alphabet; le caractère en question est transmis par paramètre.
Attention : la fonction doit traiter les caractères majuscules ainsi que les caractères minuscules !

   Astuce : Utiliser la fonction ord().

   Exemple : ALPHAPOS('A') est égal à ALPHAPOS('a') est égal à 1.

4. **(12p.)** Développez le **structogramme** du sous-programme PURE lequel retourne une chaîne de caractères. Cette chaîne est dite 'pure' ce qui veut dire, qu'elle ne contient plus de signes de ponctuations (: , . - …).
La chaîne de caractères qui est à analyser, est transmise par valeur.
La fonction CAR_PONC(C:char):boolean peut être utilisée. Elle retourne 'vrai' si le caractère passé par valeur est un signe de ponctuation.

5. **(20p.)** Développez le **structogramme** du sous-programme STRSTAT, qui sert à compter la fréquence (Häufigkeit) de tous les caractères qui composent une chaîne de caractères et stocke les fréquences dans un tableau TAB du type tTab.
```
type tTab = array [1..2, 1..26] of integer.
```

   Pour simplifier l'algorithme, on suppose que la chaîne est composée que de caractères majuscules!
La chaîne de caractères à analyser et le tableau TAB sont transmis au sous-programme.

   Les codes ASCII de l'alphabet (en majuscule) sont à écrire dans la première ligne du tableau, la deuxième ligne comporte le nombre de fois que le caractère se trouve dans la chaîne de caractères (voir

Ajournement

15-09-2003

exemple si dessous).
Attention : L'utilisation de la fonction `ALPHAPOS` (voir question 3) est obligatoire!
La chaîne à analyser ne peut être parcourue qu'une seule fois!

Exemple : chaîne de caractères à analyser : '`ALLEZLESBLEUS`'

	(A) 1	(B) 2	3	4	(E) 5	6	7	8	9	10	11	(L) 12	13	14	…	(Z) 26
1	65	66	67	68	69	70	71	72	73	74	75	76	77	78	…	90
2	1	1			3							4			…	1

▒ = des cases non adressées lors de l'exécution de STRSTAT!

Annexe : codes ASCII :

65	66	67	68	69	70	71	72	73	74	ASCII
A	B	C	D	E	F	G	H	I	J	
01	02	03	04	05	06	07	08	09	10	position

75	76	77	78	79	80	81	82	83	84	ASCII
K	L	M	N	O	P	Q	R	S	T	
11	12	13	14	15	16	17	18	19	20	position

85	86	87	88	89	90	ASCII
U	V	W	X	Y	Z	
21	22	23	24	25	26	position

97	98	99	100	101	102	103	104	105	106	ASCII
a	b	c	d	e	f	g	h	i	J	
01	02	03	04	05	06	07	08	09	10	position

107	108	109	110	111	112	113	114	115	116	ASCII
k	l	m	n	o	p	q	r	s	t	
11	12	13	14	15	16	17	18	19	20	position

117	118	119	120	121	122	ASCII
u	v	w	x	y	z	
21	22	23	24	25	26	position

- FIN -