Module Algorithmique et Programmation – Examen (durée 2 heures)

Deuxième session du 16 mai 2013

Notes : Aucun document autorisé. Sont interdits les calculatrices, les téléphones, ainsi que tout autre ustensile de calcul et/ou de communication.

Remarque 1 : Dans la suite, les indications concernant les nombres de lignes sont données en comptant toute ligne de code non-vide (#include, prototypes, accolades...).

Remarque 2 : Les petits oublis de point-virgule, parenthèse, accolades, virgules, doubles quotes, *etc.* ne seront pas pénalisant tant qu'ils restent **ponctuels**.

Remarque 3 : Les scores de chaque exercice seront cumulés pour calculer la note sur 20. Par exemple, la réussite aux exercices 1, 2 et 3 permet d'obtenir une note de 18/20, ou encore 2, 3 et 4 d'obtenir 20/20, etc. La moyenne pour réussite à l'examen est de 10/20.

Exercice 1 : Manipulation de piles (6 points)

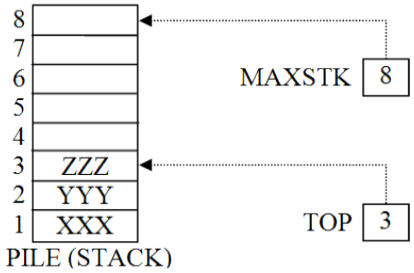
Soient les deux procédures suivantes pour la manipulation de piles :

PUSH (STACK, TOP, MAXSTK, ITEM)	**POP (STACK, TOP, ITEM)**
Si TOP = MAXSTK alors	Si TOP = 0 alors
| Print : OVERFLOW	| Print : UNDERFLOW
| Return	| Return
Fin Si	Fin Si
Faire TOP ← TOP + 1	Faire ITEM ← STACK[TOP]
Faire STACK[TOP] ← ITEM	Faire TOP ← TOP – 1
Return	Return

Souvent, TOP et MAXSTK sont des variables globales aussi ces procédures peuvent-elles être appelées en utilisant seulement :

PUSH(STACK, ITEM) et POP(STACK, ITEM)

Considérons la pile initiale ci-dessous :

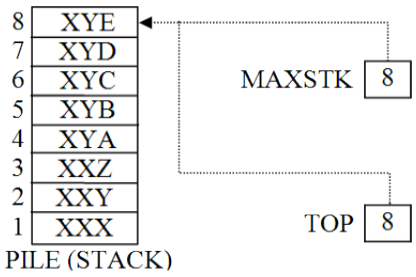


1) Simulons la séquence d'opérations suivantes :

Étape 1 : PUSH(STACK, WWW)
Étape 2 : POP(STACK, ITEM)
Étape 3 : POP(STACK, ITEM)

Donnez la représentation graphique de l'état des variables STACK, TOP, MAXSTK, ITEM à la fin de chaque étape.

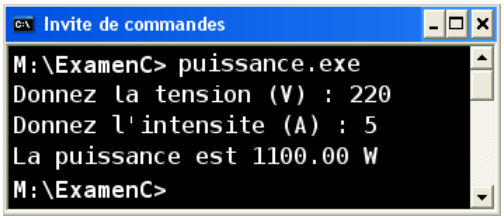
2) Reprenons la pile initiale. Décrivez le minimum d'opérations qui, en partant de cette pile, mèneront à l'état suivant :



Exercice 2 : Calcul de puissance (6 points)

La puissance électrique P en Watt se calcule à partir de la tension U en Volt et de l'intensité I en Ampère selon la formule suivante : $P = U.I$.

Écrire **UN PROGRAMME** dans lequel l'utilisateur donne au clavier la tension (en V) et l'intensité (en A), et qui affiche la puissance correspondante (en W).



Remarque : 10 lignes (environ)

Exercice 3 : Moyenne et comparaison (6 points)

Pour passer une image en noir et blanc, nous calculons la moyenne des composantes de chaque pixel. Si la moyenne est supérieure ou égale à 128, le pixel est blanc (et vaut 255), sinon il est noir (et vaut 0).

Écrire **UNE FONCTION** qui prend en paramètre les trois composantes RVB d'un pixel et qui retourne la valeur du pixel en noir et blanc. Le prototype de la fonction sera le suivant :

```
unsigned char pixel_est_gris (unsigned char R, unsigned char V, unsigned char B) ;
```

Remarque : entre 3 et 6 lignes (environ)

Exercice 4 : Chaînes de caractères et boucles (8 points)

Nous souhaitons écrire une fonction qui affiche une chaîne de caractères en remplaçant les voyelles par des points. Par exemple, si la chaîne donnée en paramètre est :

« Si vous comprenez cette phrase vous pouvez devenir developpeur »

alors l'affichage produit sera le suivant :

« S. v..s c.mpr.n.z c.tt. phr.s. v..s p..v.z d.v.n.r d.v.l.pp..r »

1) Commencer par écrire **UNE FONCTION** qui retourne vrai si le caractère passé en paramètre est une voyelle, faux sinon. Le prototype de la fonction sera le suivant :

```
int voyelle (char c) ;
```

Aide : la fonction `char tolower (char c)` retourne le caractère `c` donné en paramètre après l'avoir passé en minuscule.

Remarque : entre 3 et 10 lignes (environ)

2) Écrire **UNE FONCTION** qui prend en paramètre une chaîne de caractères et qui affiche cette chaîne, en la parcourant caractère par caractère, en remplaçant chaque voyelle rencontrée par un point. Le prototype de la fonction sera le suivant :

```
void afficher_sans_voyelles (const char *s) ;
```

Aide : la fonction `int strlen(const char *s)` retourne la longueur de la chaîne passée en paramètre.

Remarque : entre 6 et 10 lignes (environ)