Examen d'informatique 2001-2002

DEUG 1^{ere} année (MASS - MIAS - SM) - Deuxième session - Septembre 2002

Polycopiés et notes de cours autorisés - Calculatrice non autorisée

Durée : 3 heures

NUMERATION (coefficient 4)

1) Pour quelle raison la numération binaire est-elle utilisée en informatique ?

2) Que vaut le nombre décimal 1001,1 en binaire ? Si le résultat a un développement infini périodique, donnez son développement jusqu'à la première période incluse.

3) Codez le nombre décimal -12 en complément à 1 sur 7 bits, puis en complément à 2 sur 8 bits.

4) Effectuez la multiplication $(2120)_3 \times (1021)_3$.

ARCHITECTURE DES ORDINATEURS (coefficient 3)

1) Une image noir et blanc est formée de 400×300 pixels, chaque pixel étant codé sur 1 octet. Quel est l'espace mémoire nécessaire pour stocker cette image ? Expliquez le procédé à suivre pour convertir ce nombre en Ko. Donnez une valeur approchée de votre résultat en Ko.

2) Combien de couleurs peut-on coder sur 3 octets ?

3) Combien de bits seraient nécessaires pour adresser une mémoire contenant 4 Gmots (gigamots) ? Justifiez votre réponse.

PROGRAMMATION (coefficient 3)

Ecrire un programme en PASCAL qui propose de calculer selon le choix de l'utilisateur :

- la somme,
- la différence,
- le produit,
- le quotient

de deux nombres entrés par l'intermédaire du clavier, puis affiche le résultat et le type d'opération arithmétique effectuée.

COMPREHENSION (coefficient 2)

Imaginez que vous exécutiez le programme suivant :

```
program examen ;
var i, n : integer ;
      x, y1, y2, y3 : real ;
begin
      readln(n) ;
      readln(x) ;
      y1 := x ;
      for i := 1 to n do
            y1 := y1 * x ;
      writeln('y1=', y1) ;
      i := 1 ;
      y2 := 1 ;
      while (i < n) do
      begin
            y2 := y2 * x ;
            i := i + 1 ;
      end;
      writeln('y2=', y2) ;
      i := 1 ;
      y3 := 1 ;
      repeat
            y3 := y3 * x ;
            i := i + 1 ;
      until (i > n) ;
      writeln('y3=', y3) ;
end.
```

De façon générale, à quoi correspondent les valeurs de y1, y2 et y3 calculées par le programme ? Donnez les valeurs de y1, y2 et y3 affichées par ce programme dans le cas où x = 3 et n = 2.

PROBLEME : recherche de palindrome (coefficient 8)

Il vous est demandé d'écrire un programme permettant, à partir de N lettres saisies au clavier, d'afficher l'ensemble des "mots" formés à partir de ces N lettres, puis de rechercher les éventuels palindromes (mot pouvant être lu indifféremment de gauche à droite ou de droite à gauche, par exemple : *radar*) parmi l'ensemble de ces "mots".

Pour la programmation, vous vous limiterez au cas $N = 3$.

- Exemple 1 : si les trois lettres entrées au clavier sont a, b et c, le programme affichera les "mots" : **abc** ; **acb** ; **bac** ; **bca** ; **cab** ; **cba** et précisera qu'aucun de ces "mots" n'est un palindrome.
- Exemple 2 : si les trois lettres entrées au clavier sont a, b et a, le programme affichera les "mots" : **aba** ; **aab** ; **baa** ; **baa** ; **aab** ; **aba** et précisera que le "mot" **aba** est un palindrome.

Remarque sur la façon d'obtenir tous les "mots" possibles à partir des 3 lettres $L[1]$, $L[2]$ et $L[3]$:

- pour la première lettre du "mot", il y a 3 choix possibles : $L[1]$, $L[2]$ ou $L[3]$;
- pour la seconde lettre du "mot", il n'y a que 2 choix possibles : par exemple $L[2]$ ou $L[3]$ si $L[1]$ a été choisie comme première lettre ;
- pour la troisième lettre du "mot" il n'y qu'un seul choix : par exemple $L[3]$ si $L[1]$ et $L[2]$ ont été choisies comme première et seconde lettres ;

ainsi, pour 3 lettres, il y a $3 \times 2 \times 1 = 6$ "mots" dont certains peuvent être identiques si certaines des lettres $L[1]$, $L[2]$ et $L[3]$ sont identiques (voir Exemple 2). Les "mots" seront formés par concaténation (assemblage de lettres).

Les tâches que vous devez programmer en PASCAL sont énoncées dans un ordre de difficulté croissante. Il est donc conseillé de suivre pas à pas les étapes proposées.

Etape 1

Il vous est demandé d'écrire un premier programme **Prog1**, sans recourir à l'usage de procédures et de fonctions, effectuant les tâches suivantes :
a) déclaration d'un tableau de caractères L et d'un tableau de chaînes de caractères M ;
b) remplissage du tableau L avec $N = 3$ lettres saisies au clavier ;
c) affichage des "mots" formés par concaténation à partir des 3 lettres stockées dans le tableau L (utilisez pour cela des boucles *for* imbriquées), les "mots" seront stockés dans le tableau M ;
d) recherche et affichage des éventuels palindromes parmi l'ensemble des "mots" stockés dans le tableau M.

Etape 2

Vous allez maintenant écrire un programme effectuant des tâches similaires mais que vous allez structurer en introduisant des procédures et fonctions :
a) écrire la procédure **Lecture** de saisie des 3 lettres au clavier et de stockage dans le tableau L (les arguments de la procédure étant N et L) ;
b) écrire la procédure **Mots** qui forme toutes les combinaisons de 3 lettres, les stocke dans le tableau M et les affiche (les arguments de la procédure étant N, L et M) ;
c) écrire la fonction booléenne **Palindrome** qui retourne la valeur **true** si le i^{eme} "mot" du tableau M est un palindrome ou bien la valeur booléenne **false** si le i^{eme} "mot" du tableau M n'est pas un palindrome (les arguments de la fonction étant N, M et l'indice i) ;
d) écrire le programme principal **Prog2** qui fait appel aux procédures **Lecture** et **Mots** et à la fonction **Palindrome** (le programme fera appel à cette fonction pour chacun des "mots" stockés dans M).

Etape 3 : optimisation de votre programme

Modifiez la procédure **Mots** afin d'éviter le stockage et l'affichage multiple d'un même "mot".