Nom : ..... Prénom : .....

1B1 Devoir d'informatique II,2 27.04.2017

Partie théorique (30 points - 45 min)

Question 1 — 10 (=6+4) points

(1) Ecrire la fonction récursive **`puissRapid`** qui calcule de manière rapide x^n, où x est un ***nombre réel*** et n est un entier ***positif*** ou ***négatif***. On demande d'indiquer tous les cas où la fonction retourne un résultat faux.

(2) En précisant toutes les étapes, montrer comment cette fonction calcule $(-0,02)^{-4}$.

Question 2 — 7 (=3+4) points

(1) Ecrire la fonction **`factorielle`** qui calcule ***récursivement*** $n!$ pour un ***entier naturel*** n donné. (On rappelle que par convention : $0! = 1! = 1$)

(2) En déduire la fonction **`binomial`** qui prend en entrée deux entiers n et p et qui retourne le coefficient binomial C_n^p (c.-à-d. le nombre de combinaisons de n objets pris p à p). On rappelle que :

$$C_n^p = \frac{n!}{p!(n-p)!}, \text{ si } 0 \le p \le n \text{ et } n \ge 0$$

La fonction **`binomial`** doit retourner 0 si l'une (au moins) des conditions d'existence de cette définition n'est pas vérifiée.

Question 3 — 13 (=3+9+1) points

On donne la procédure suivante d'une application en console :

```
procedure bubble(var s:string;d:integer);
var i:integer;
begin
  if d>1 then
    begin
      for i:=1 to d-1 do
        if s[i]>s[i+1] then
         begin
           echange(s,i,i+1);
           writeln(s)
         end;
      bubble(s,d-1)
    end;
end;
```

(1) Ecrire la procédure **`echange`** qui apparaît dans la procédure **`bubble`** et qui permet d'échanger les caractères d'indices **`i`** et **`j`** donnés d'un string **`s`**.

(2) On veut exécuter la procédure bubble avec **`s:='mathe'`** et **`d:=5`**.
   a) Ecrire l'appel correspondant.
   b) Reproduire l'écran de sortie généré par cet appel.

(3) On enlève maintenant la ligne contenant writeln de la procédure **`bubble`** et on définit un string **`t:='dcbaaaa'`**. Ecrire une seule instruction qui permet de transformer ce string en **`t:='abcdaaa'`** à l'aide de la procédure.

Partie pratique : Courbes de Lissajous (30 points - 75 min)

Le but de cette partie est de représenter graphiquement les courbes de Lissajous définies par leurs équations paramétriques :

$$\begin{cases} x(t) = \sin(p \cdot t) \\ y(t) = \sin(q \cdot t + \varphi) \end{cases} \quad \text{avec } 0 \le t \le 2\pi$$

où p et q sont deux entiers et φ (phi) est un réel dans [0,2π], appelé le déphasage.

Remarques : a) Comme $-1 \le \sin\alpha \le 1$, les coordonnées réelles varieront toujours entre $x_{min} = y_{min} = -1$ et $x_{max} = y_{max} = 1$. b) On recommande d'utiliser des variables globales pour stocker les valeurs de p, q, φ , … au choix du candidat.

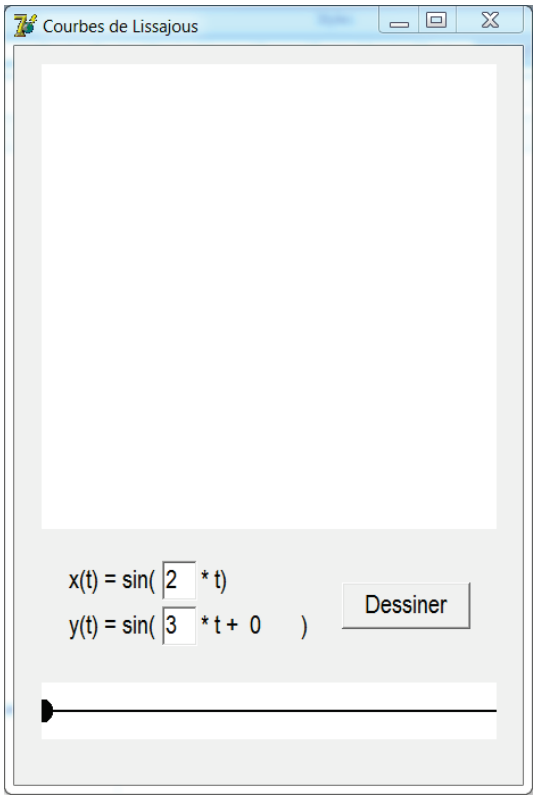

fig. 1 (au démarrage du programme)

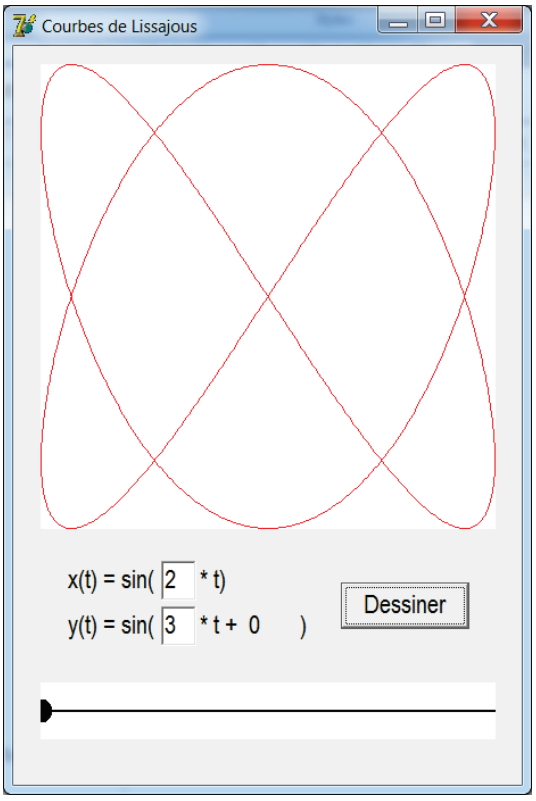

fig.2 (après un clic sur Dessiner)

(1) Créer l'interface graphique de la *figure 1* comportant un bouton **btnDessiner** et deux images : **imgDessin**, carrée (c.-à-d. de largeur et de hauteur égales), qui contiendra le graphique et **imgSlider** qui contiendra un curseur pour faire varier le paramètre φ. L'utilisateur pourra écrire les paramètres p et q dans deux boîtes d'éditions **edtP** et **edtQ** (valeurs par défaut 2 et 3 respectivement). Le paramètre φ se trouve dans un label **lblPhi** (valeur par défaut 0). Sa valeur pourra uniquement être modifiée en cliquant ou en déplaçant la souris sur l'image **imgSlider** (cf. question 6). ***(4 points)***

(2) Ecrire la procédure **dessinerCurseur** qui prend en entrée deux paramètres : l'image **imgSlider** et un réel **phi**, et qui a) efface l'image **imgSlider** puis b) y dessine le curseur (un disque noir de rayon 10 pixels sur un segment de droite d'épaisseur 2 pixels traversant l'image au milieu, de gauche à droite) à l'endroit déterminé par le réel **phi** : **phi** peut varier entre 0 (auquel cas le centre du disque se trouve à l'extrême gauche de l'image, cf. *fig. 1*), et 2π (auquel cas il se trouve à l'extrême droite de l'image). ***(7 points)***

(3) Ecrire la méthode appelée par l'événement **OnCreate** du formulaire. Cette procédure permettra d'initialiser éventuellement certaines variables globales, fera apparaître en blanc le canevas de **imgDessin**, et appellera la procédure **dessinerCurseur** pour représenter le curseur sur l'image **imgSlider** à la position φ = 0. (voir *fig. 1*). ***(2 points)***

(4) Ecrire une procédure **dessinerLissajous** qui prend en entrée 4 paramètres : l'image **imgDessin**, deux entiers **p** et **q** et un réel **phi**, et qui a) efface l'image **imgDessin** puis b) y dessine en rouge la courbe de Lissajous avec les paramètres **p**, **q** et **phi**. Pour cela on fera varier le temps t de 0 à 2π en 1000 pas :

$$t_n = 2\pi n / 1000 \text{ , avec } 0 \le n \le 1000$$

Pour chaque t_n on calculera les coordonnées réelles $x(t_n)$ et $y(t_n)$, on transformera ces coordonnées réelles en pixels en tenant compte de la ***remarque a)*** et on reliera les points correspondants sur le graphique par des segments de droite. ***(8 points)***

(5) Un clic sur le bouton ***Dessiner*** lira les valeurs des paramètres *p*, *q* et φ dans les boîtes d'édition et le label correspondants et appellera la procédure **dessinerLissajous** pour dessiner la courbe de Lissajous correspondante. ***(2 points)***

(6) Ecrire les procédures relatives aux événements **MouseDown** et **MouseMove** de l'image **imgSlider** : a) Lorsque l'utilisateur clique avec le bouton gauche sur cette image, le curseur (c.-à-d. le disque noir) viendra se placer à l'abscisse de la position de la souris, la valeur du paramètre **phi** sera calculée en fonction de cette position du curseur, le label **lblPhi** sera actualisé et la nouvelle courbe de Lissajous avec les paramètres *p*, *q* et φ sera dessinée sur **imgDessin** (cf. *fig. 3*). b) Exactement la même chose devra se produire continuellement lorsque la souris se déplacera sur l'image **imgSlider** avec le bouton gauche enfoncé. ***(7 points)***

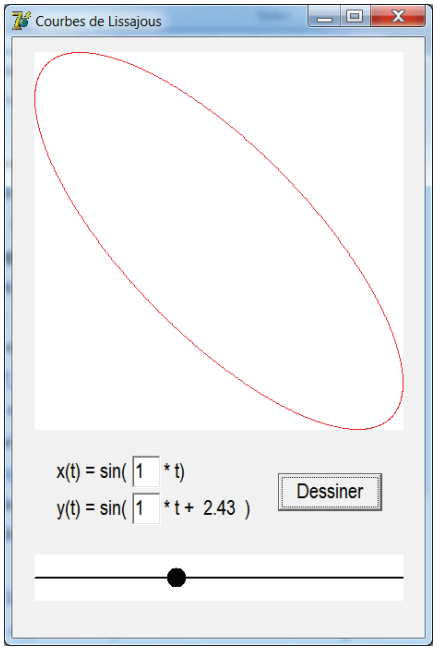

figure 3