Exercices d'informatique - Série n° 4

Cours 3INOC01 Série distribuée le 9.1.2017

1. Le module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
1  import Tkinter
2  racine=Tkinter.Tk()
3  racine.title("Exercice pour le module Tkinter")#Modifier le texte
4  racine.geometry("600x300+2048+2000")#Modifier les quatre nombres
5  print racine.maxsize()
6  print racine.minsize()
7  racine.resizable(0,0)#Remplacer (0,0) par (0,1), (1,0) et (1,1)
8  racine.mainloop()
```

2. La classe *Label* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
1  import Tkinter
2  cadre=Tkinter.Tk()
3  cadre.title("Exercice pour le module Tkinter")
4  a=cadre.maxsize()
5  cadre.geometry(str(a[0])+"x"+str(a[1]/2)+"+0+"+str(a[1]))
6  petit_texte=Tkinter.Label(text="Bonjour, voici un petit\n texte",
7                            background="blue",
8                            foreground="#ff1100",#Couleur sous la forme RRVVBB
9                            #Choix possibles pour R,V et B: 0,1,2,3,4,5,6,7,8,9,a,b,c,d,e,f
10                           borderwidth=10,
11                           padx=100,
12                           pady=10,
13                           font=("Helvetica", 24),
14                           wraplength=500,
15                           height=2,
16                           width=20,
17                           anchor="nw",
18                           relief="raised"#Essayer: sunken, flat, ridge, solid et groove
19                           )
20 petit_texte.place(anchor="nw",relheight=0.5,relwidth=0.4,relx=0.01,rely=0.2)
21 #Essayer des autres nombres compris entre 0 et 1
22 cadre.mainloop()
```

3. La classe *Label* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
1  import Tkinter
2  exercice=Tkinter.Tk()
3  exercice.title("Exercice pour le module Tkinter")
4  exercice.geometry("1000x400")
5  dessin=Tkinter.PhotoImage(file="essai.gif")
6  petite_image=Tkinter.Label(image=dessin)
7  petite_image.place(anchor="nw",height=300,width=500,x=200,y=50)
8  exercice.mainloop()
```

4. Le *Packer*.

Exécuter le script suivant et déterminer l'action de chaque ligne.

```
1  import Tkinter
2  cadre=Tkinter.Tk()
3  cadre.geometry("600x400")
4  T=Tkinter.Label(text='Bonjour',background='red',relief='raised')
5  T.pack()
6  #T.pack(side='top', fill='x')#Remplacer x par y ou both et left par right, bottom ou top
7  Tkinter.mainloop()
```

5. La classe *Text*.

Exécuter le script suivant et déterminer l'action de chaque ligne.

```
1  import Tkinter
2  cadre=Tkinter.Tk()
3  cadre.geometry("600x400")
4  T=Tkinter.Text()
5  T.pack(side='left', fill='y')
6  T.insert('end','Bonjour')
7  print T.get('1.1','1.4')
8  print T.search('jou','1.0')#nocase=False
9  T.delete('1.3','1.4')
10 Tkinter.mainloop()
```

6. La classe *Button* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
1  import Tkinter
2  racine=Tkinter.Tk()
3  racine.geometry("1000x300")
4  bouton=Tkinter.Button(racine, text="Quitter",
5                        command=racine.quit,
6                        background='red',
7                        activebackground='magenta',
8                        activeforeground='yellow')
9  bouton.place(x=200,y=50)
10 racine.mainloop()
```

7. La classe *Button* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
1  import Tkinter
2  racine=Tkinter.Tk()
3  racine.geometry("1000x300")
4  dessin=Tkinter.PhotoImage(file="Smiley.gif")
5  bouton=Tkinter.Button(racine, image=dessin,
6                        command=racine.quit,
7                        background='red',
8                        activebackground='magenta')
9  bouton.place(x=200,y=50)
10 racine.mainloop()
```

8. La classe *Button* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
import Tkinter
def afficher():
    texte=Tkinter.Label(text="Bravo !",background='blue')
    texte.place(x=400,y=100)
racine=Tkinter.Tk()
racine.geometry("1000x400")
bouton1=Tkinter.Button(racine, text="Quitter",
                       command=racine.quit,
                       background='red',
                       activebackground='magenta')
bouton2=Tkinter.Button(racine, text="Afficher",
                       command=afficher,
                       background='green',
                       activebackground='yellow')
bouton1.place(x=200,y=50)
bouton2.place(x=150,y=100)
racine.mainloop()
```

9. La classe *Checkbutton* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
def afficher():
    texte="Case: "+str(case.get())
    Tkinter.Label(text=texte).place( x=400,y=100)
import Tkinter
exercice=Tkinter.Tk()
exercice.geometry("1000x400")
case=Tkinter.IntVar()
bouton=Tkinter.Checkbutton(exercice,
                           text="Une petite case: ",
                           variable=case,
                           command=afficher)
bouton.place(x=100,y=20)
afficher()
#bouton.select()#Remplacer .select par .deselect ou invoke ou toggle
exercice.mainloop()
```

10. La classe *Radiobutton* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
def afficher1():
    texte="Groupe 1: "+str(groupe1.get())
    Tkinter.Label(text=texte).place(x=400,y=100)
def afficher2():
    texte="Groupe 2: "+groupe2.get()
    Tkinter.Label(text=texte).place(x=400,y=150)
import Tkinter
exercice=Tkinter.Tk()
exercice.geometry("1000x400")
groupe1=Tkinter.IntVar()
bouton1=Tkinter.Radiobutton(exercice,
                            text="1: ",
                            variable=groupe1,
                            value=1,
                            command=afficher1)
bouton2=Tkinter.Radiobutton(exercice,
                            text="2: ",
                            variable=groupe1,
                            value=2,
                            command=afficher1)
```

```
bouton3=Tkinter.Radiobutton(exercice,
                            text="3: ",
                            variable=groupe1,
                            value=3,
                            command=afficher1)
bouton1.place(x=100,y=20)
bouton2.place(x=150,y=20)
bouton3.place(x=200,y=20)
bouton2.select()
afficher1()
groupe2=Tkinter.StringVar()
boutona=Tkinter.Radiobutton(exercice,
                            text="A: ",
                            variable=groupe2,
                            value="moi   ",
                            command=afficher2)
boutonb=Tkinter.Radiobutton(exercice,
                            text="B: ",
                            variable=groupe2,
                            value="toi   ",
                            command=afficher2)
boutonc=Tkinter.Radiobutton(exercice,
                            text="C: ",
                            variable=groupe2,
                            value="nous  ",
                            command=afficher2)
boutona.place(x=100,y=50)
boutonb.place(x=150,y=50)
boutonc.place(x=200,y=50)
boutonc.select()
afficher2()
exercice.mainloop()
```

11. La classe *Listbox* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
def afficher():
    texte=liste.get(0,"end")#Remplacer 0 et "end" par 2 et 3 ou un seul nombre
    Tkinter.Label(text=texte).place(x=400,y=100)
    texte2=str(liste.curselection())+str(type(liste.curselection()))
    Tkinter.Label(text=texte2).place(x=400,y=150)
    liste.select_clear(1,None)
    liste.delete(3,4)#Remplacer 3 et 4 par des autres nombres ou "end" ou un seul nombre
import Tkinter
bernard=Tkinter.Tk()
bernard.geometry("1000x400")
choix=['Bernard','Paul','Pierre','Thomas','Mathilde']
liste=Tkinter.Listbox(height=3,
                      selectmode="single")#Remplacer single par browse, multiple, extended
bouton=Tkinter.Button(text='Go !',
                      command=afficher)
for element in choix:
    liste.insert("end",element)
liste.select_set(1,None)
liste.place(x=100,y=20)
bouton.place(x=250,y=20)
bernard.mainloop()
```

12. La classe *Scale* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
import Tkinter
def afficher(bernard):
```

```
3       Tkinter.Label(text=str(bernard)+10*" ").place(x=300,y=10)
4       print(donnee.get())
5   cadre=Tkinter.Tk()
6   cadre.geometry("600x100")
7   donnee=Tkinter.Scale(cadre,
8                        orient="horizontal",#Remplacer par "vertical"
9                        label="Choisir:",
10                       length=300,
11                       troughcolor='red',
12                       sliderlength=10,
13                       from_=-12,
14                       to=12,
15                       tickinterval=2,
16                       command=afficher)
17  donnee.place(x=0,y=0)
18  donnee.set(-6)
19  cadre.mainloop()
```

13. La classe *Scrollbar*.

Exécuter le script suivant et déterminer l'action de chaque ligne.

```
1   import Tkinter
2   racine=Tkinter.Tk()
3   s =Tkinter.Scrollbar()
4   T = Tkinter.Text()
5   s.config(command=T.yview)
6   T.config(yscrollcommand=s.set)
7   s.pack(side='right', fill='y')
8   T.pack(side='left', fill='y')
9   racine.mainloop()
```

14. La classe *Canvas* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
1   import Tkinter
2   cadre=Tkinter.Tk()
3   cadre.geometry("600x400")
4   a=Tkinter.Canvas()
5   bernard=a.create_line(19,12,15,26,33,55,120,15,250,200,
6                         width=1,
7                         fill='red',
8                         arrow="first",#Remplacer first par none, last, both
9                         smooth="true",#Remplacer true par false
10                        tags='1')
11  a.itemconfig(bernard,fill='blue')
12  a.place(x=10,y=10)
13  print a.bbox(bernard)
14  print a.gettags(bernard)
15  print a.itemcget(bernard,"width")
16  #a.delete(bernard)
17  #a.coords(bernard,100,100,20,20)
18  cadre.mainloop()
```

15. La classe *Canvas* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
1   import Tkinter
2   cadre=Tkinter.Tk()
3   cadre.geometry("600x400")
4   a=Tkinter.Canvas()
5   gustave=a.create_polygon(0,150,50,100,0,50,50,0,
6                            fill='red',
```

```
                      outline='yellow',
                      smooth="false",#Remplacer true par false
                      tags='t1',
                      width=3)
a.place(x=10,y=10)
cadre.mainloop()
```

16. La classe *Canvas* du module *Tkinter*.

Déterminer l'action de la méthode create_oval du *widget Canvas* et le rôle des paramètres width et fill en exécutant le script suivant:

```
import Tkinter
cadre=Tkinter.Tk()
cadre.geometry("600x450")
a=Tkinter.Canvas(height=350,width=450)
bernard=a.create_oval(10,10,50,50,
                      width=1,
                      fill='red')
a.place(x=10,y=10)
cadre.mainloop()
```

17. La classe *Canvas* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
import Tkinter
cadre=Tkinter.Tk()
cadre.geometry("600x400")
a=Tkinter.Canvas()
gustave=a.create_rectangle(0,150,50,100,
                      fill='red',
                      outline='yellow',
                      tags='t1',
                      width=3)
a.place(x=10,y=10)
cadre.mainloop()
```

18. La classe *Canvas* du module *Tkinter*.

Exécuter le script suivant et déterminer l'action de chaque instruction.

```
import Tkinter
cadre=Tkinter.Tk()
cadre.geometry("600x400")
a=Tkinter.Canvas()
gustave=a.create_text(78,150,
                      anchor='n',#Remplacer n par s, e, w, ne, nw, se, sw ou center
                      fill='red',
                      font='helvetica 24',
                      tags='t1',
                      text='Bonjour')
a.place(x=10,y=10)
cadre.mainloop()
```

19. La méthode *after* des *widget*.

Exécuter le script suivant et déterminer l'action de la méthode after.

```
import Tkinter
def compteur(n):
    texte.config(text="Temps: "+str(n)+" secondes")
cadre=Tkinter.Tk()
cadre.geometry("300x100")
texte=Tkinter.Label(text="Temps: ")
```

```
texte.place(x=50,y=50)
time=2000#Remplacer 2000 par 10000
id=cadre.after(time,compteur,time/1000)#Remplacer cadre par texte
print id
#cadre.after_cancel(id)
cadre.mainloop()
```

20. La méthode *after* des *widget*.

Rédiger un script qui affiche un compte à rebours de 200 à 0 dans une fenêtre (comme sur la figure 1) et qui s'effectue en environ 20 secondes.

FIGURE 1. Exercice 20

21. Liaisons de rappels aux événements.

Déterminer l'action de la méthode bind en exécutant le script suivant:

```
import Tkinter
cadre=Tkinter.Tk()
cadre.geometry("500x100")
def affichage(*ignore):
    texte.config(text="BRAVO ... !")
texte=Tkinter.Label(text="Pressez sur la touche RETURN ! ")
texte.place(x=50,y=30)
cadre.bind('<Return>',affichage)
cadre.mainloop()
```

22. Liaisons de rappels aux événements.

Déterminer l'action de la méthode bind en exécutant le script suivant:

```
# -*- coding: utf-8 -*-
import Tkinter
cadre=Tkinter.Tk()
cadre.geometry("500x100")
def affichage(evt):
    if evt.char==evt.keysym:
        msg="Touche normale: "+evt.char
    elif len(evt.char)==1:
        msg="Touche de ponctuation: "+evt.char+" (nom: "+evt.keysym+")"
    else:
        msg="Touche sp'eciale: "+evt.keysym
    texte.config(text=msg)
    #cadre.unbind('<Key>')
texte=Tkinter.Label(text="Pressez une touche du clavier: ")
texte.place(x=50,y=30)
cadre.bind('<Key>',affichage)
cadre.mainloop()
```

23. Jeu de la touche interdite.

Créer un jeu où l'utilisateur gagne un point à chaque fois qu'il presse une nouvelle touche (presser deux fois la même touche ne donne qu'un seul point) différente de la touche inconnue contenant la "bombe" qui arrête le jeu (voir figure 2) !

FIGURE 2. Exercice 23: la bombe est sous la touche "g".

24. Evénements souris.

Exécuter le script suivant et déterminer l'action de chaque ligne du code.

```
1  # -*- coding: utf-8 -*-
2  import Tkinter
3  cadre=Tkinter.Tk()
4  cadre.geometry("600x400")
5  texte="Cliquez sur les boutons de la souris en la d'eplac_cant !"
6  L=Tkinter.Label(cadre,text=texte)
7  R=Tkinter.Label(cadre,text='')
8  E=Tkinter.Entry(cadre)
9  L.place(x=20,y=20)
10 R.place(x=20,y=50)
11 E.place(x=20,y=80)
12 def souris(nomEvt):
13     def affichage(evt):
14         R.config(text="Ev'enement souris: "+nomEvt+' x: '+str(evt.x)+' y: '+str(evt.y))
15     cadre.bind_all(nomEvt,affichage)#Remplacer bind_all par bind et cadre par E ou L
16 souris('<Button-1>')
17 souris('<Button-2>')
18 souris('<Button-3>')
19 souris('<ButtonRelease-1>')
20 souris('<ButtonRelease-2>')
21 souris('<ButtonRelease-3>')
22 souris('<Double-Button-1>')
23 souris('<Double-Button-2>')
24 souris('<Double-Button-3>')
25 souris('<B1-Motion>')
26 souris('<B2-Motion>')
27 souris('<B3-Motion>')
28 souris('<Motion>')
29 souris('<Enter>')
30 souris('<Leave>')
31 cadre.mainloop()
```

25. Evénements souris.

Rédiger un script qui affiche un texte dont la couleur change lorsque la souris passe dessus comme sur la figure 3.

26. Evénements souris.

(1) Rédiger un script qui affiche un texte dont la position peut être modifiée en déplaçant la souris tout en maintenant le bouton gauche appuyé comme sur la figure 4. Attention, le texte doit se déplacer aussi si l'utilisateur presse le bouton gauche lorsque la souris n'est pas sur le texte.

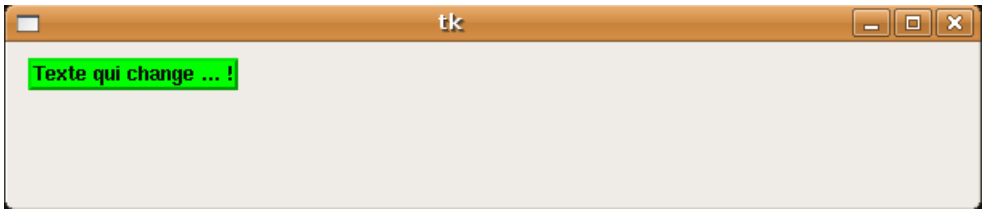

FIGURE 3. Exercice 25

FIGURE 4. Exercice 26

(2) Rédiger un deuxième script où le texte se déplace seulement si la souris est sur le texte quand l'utilisateur la déplace en maintenant le bouton gauche appuyé. Que constate-on ?

27. Evénements souris.
Rédiger un script qui affiche un point rouge dont la position peut être modifiée en déplaçant la souris tout en maintenant le bouton gauche appuyé comme sur la figure 5.

28. Evénements souris.
Rédiger un script permettant de faire des dessins avec la souris comme sur la figure 6.

29. Dessins.
Rédiger un script *Python* grâce auquel l'utilisateur peut placer dans la fenêtre des petits ronds ou des petits carrés, rouges, verts ou bleus en appuyant sur les touches d (rond) ou c (carré) pour choisir la forme et en appuyant sur les touches r (rouge), g (vert), b ou (bleu) pour choisir la couleur puis en cliquant sur le bouton gauche de sa souris à l'endroit où il désire placer son objet (voir figure 7). Le choix de la forme ou de la couleur ne doit pas être répété avant chaque "clic" de souris.

30. La classe *Canvas* du module *Tkinter*.
Rédiger un script qui affiche une fenêtre (comme sur la figure 8) contenant une courbe fermée, dont la couleur, un des points et la régularité peuvent être modifiés. Ajouter un bouton permettant de placer une flèche sur la courbe.

31. La classe *Canvas* du module *Tkinter*.
Rédiger un script qui affiche une fenêtre (comme sur la figure 9) contenant un polygone, un rectangle, un texte et un bouton permettant de modifier les couleurs de ces trois objets.

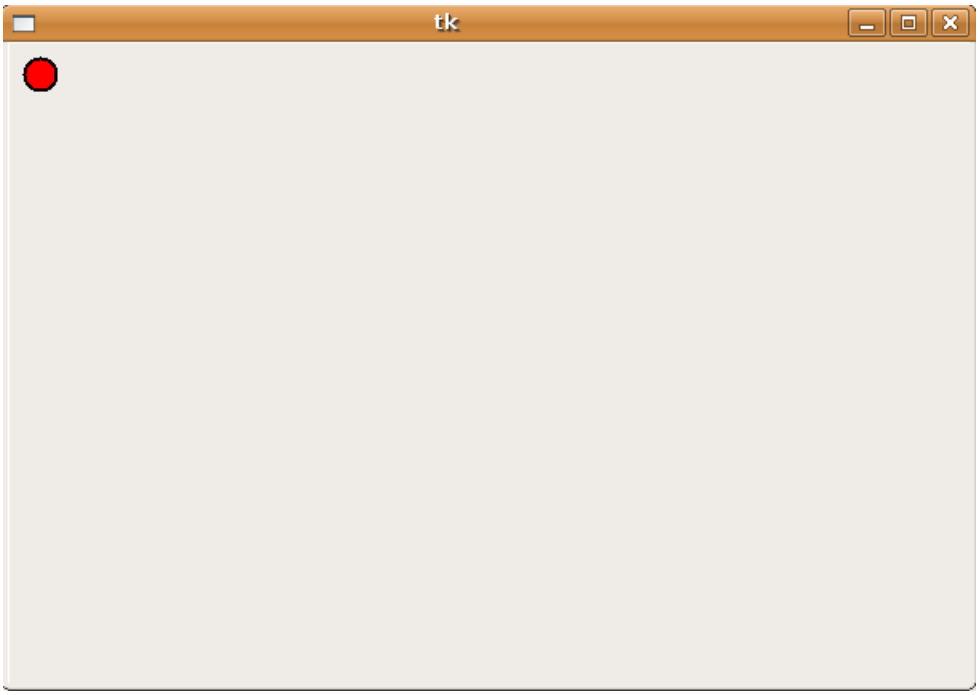

FIGURE 5. Exercice 27

FIGURE 6. Exercice 28

32. La classe *Canvas* du module *Tkinter*.
Rédiger un script qui affiche une fenêtre (comme sur la figure 10) contenant un ovale rouge, dont la couleur et la taille verticale peuvent être modifiées.

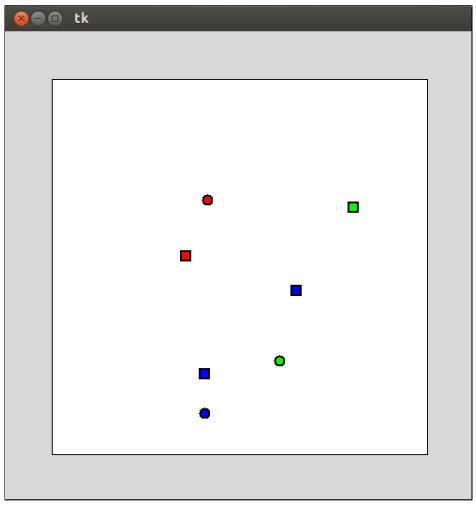

FIGURE 7. Exercice 29

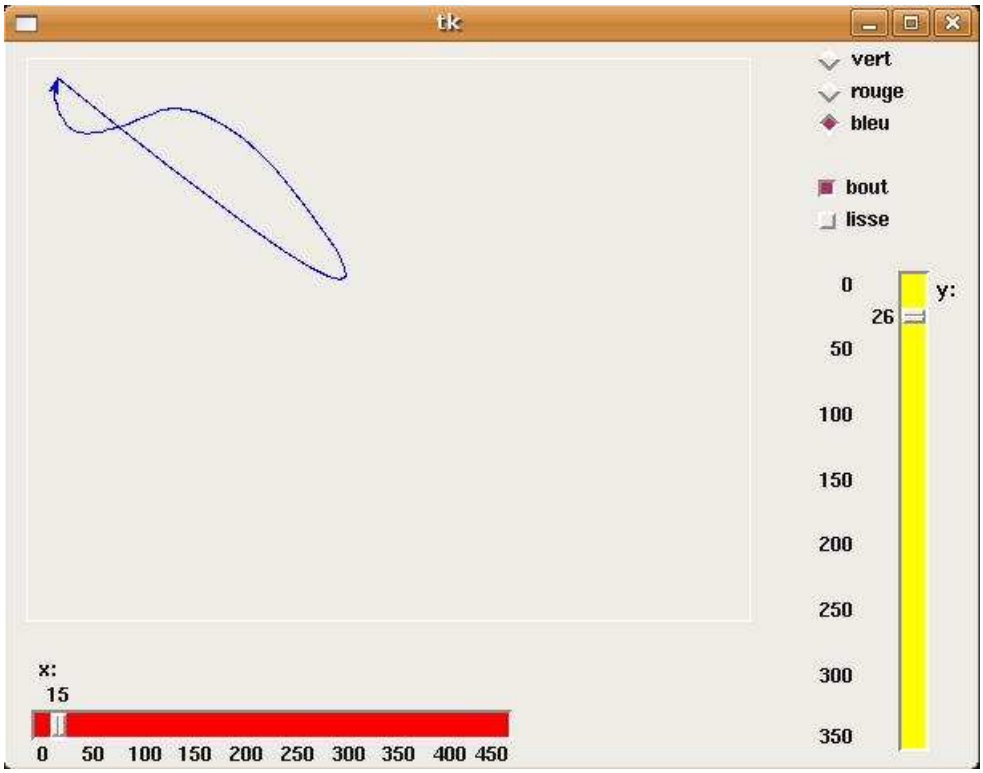

FIGURE 8. Exercice 30

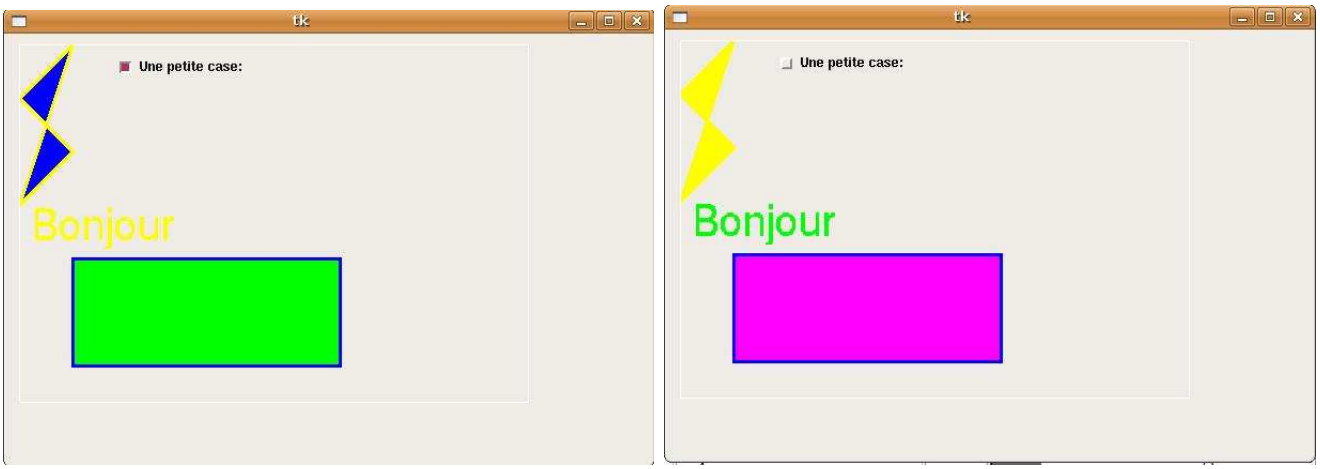

FIGURE 9. Exercice 31

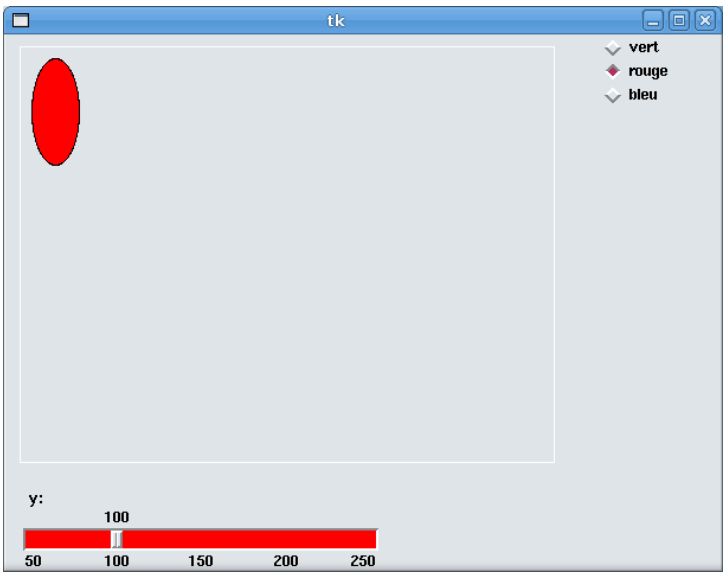

FIGURE 10. Exercice 32