Тема 14. Декілька прикладів. Гра «життя»

Декілька прикладів

Гра «життя»

(за https://habrahabr.ru/company/mailru/blog/228379/)

```
з tinter import Tk, Canvas, Button, Frame, BOTH, NORMAL, HIDDEN
# функція отримує координати миші # в момент натискання або пересування із затиснутою кнопкою
defdraw_a(e):
     ii = (ey-3) / cell_size
     jj = (ex-3) / cell_size
     ii = int (ii); jj = int (jj)
     print("e=",e, 'ii=',ii, 'jj=',jj)
     canvas.itemconfig(cell_matrix[addr(ii, jj)], state=NORMAL, tags='vis')
```

ця функція перетворює двовимірну координату в простий адресу одномірного масиву

```
defaddr(ii,jj):
     if(ii < 0 or jj < 0 or ii >= field_height or jj >= field_width):
          return int( len(cell_matrix)-1 )
     else:
          return int( ii * (win_width / cell_size) + jj )


# функція оновлення «картинки»
defrefresh():
     fori in range(field_height):
          forj in range(field_width):
               k = 0
              # вважаємо число сусідів
               fori_shift in range(-1, 2):
                    forj_shift in range(-1, 2):
                         if(canvas.gettags( cell_matrix[addr(i + i_shift, j + j_shift)])[0] == 'vis' and (i_shift != 0 або j_shift != 0)):



                              k += 1
               current_tag = canvas.gettags(cell_matrix[addr(i, j)])[0]

               # в залежності від їх числа встановлюємо стан клітини
```

```
                тут можна експериментувати з самими
"правилами" гри
                if(k == 3):
                     canvas.itemconfig(cell_matrix[addr(i, j)],
tags=(current_tag, 'to_vis'))

                if(k <= 1 або k > = 4):
                     canvas.itemconfig(cell_matrix[addr(i, j)],
tags=(current_tag, 'to_hid'))

                if(k == 2 and canvas.gettags(cell_matrix[ addr(i,
j)])[0] == 'vis'):


                    canvas.itemconfig(cell_matrix[addr(i, j)],
tags=(current_tag, 'to_vis'))


сам крок: оновлюємо стан і малюємо
defstep():
     refresh()
     repaint()

defclear():
     for i in range(field_height):
          for j in range(field_width):
               canvas.itemconfig(cell_matrix[addr(i, j)],
state=HIDDEN, tags=('hid','0'))


# перемальовуємо поле по буферу # згідно з другим тегом елемента
defrepaint():
     fori in range(field_height):
          for jinrange(field_width):
               if(canvas.gettags(cell_matrix[addr(i, j)])[1] ==
'to_hid'):

                    canvas.itemconfig(cell_matrix[addr(i, j)],
state=HIDDEN, tags=('hid','0'))

               if(canvas.gettags(cell_matrix[addr(i, j)])[1] ==
'to_vis'):

                    canvas.itemconfig(cell_matrix[addr(i, j)],
state=NORMAL, tags=('vis','0'))


# «створення та підтримка виконання» # автоматичного
режиму

defloop():
```

```
    ##label['text'] = str(n)
    step()
    if (flag_run):
            root.after(500, loop) # call loop() in 0.5 seconds

# «Виконання» автоматичного режиму
defrun():
        global flag_run, btn3
        #print('flag_run=',flag_run)
        if not flag_run:
                btn3.config(text="STOP")
                flag_run=True
                #print('stop')
                loop()
        else:
                btn3.config(text="RUN")
                flag_run=False
                #print('run')

flag_run=False # працює «автоматично» або по кроках

# створюємо саме вікно
root = Tk()
root.title('4 cuba')
# це ширина та висота вікна
win_width = 360 # 350
win_height = 380 # 370
# «будуємо» конфігураційний рядок – розмір вікна
config_string = "{0}x{1}".format(win_width, win_height + 32)
# Задаємо колір клітини
fill_color = "green"
методом geometry() задаємо розміри, можна написати і
просто рядок виду '360x380'
root.geometry(config_string)
це ширина самої клітини, з урахуванням «просвіту»
cell_size = 20
# створюється об'єкт типу Canvas, на якому буде
відбуватися безпосередньо малювання,

# він робиться дочірнім по відношенню до самого вікна
root
canvas = Canvas (root, height = win_height)
# пакуємо його, аналог show() в інших системах
canvas.pack(fill=BOTH)
# Визначаємо розміри поля в клітинах
field_height = int( win_height / cell_size )
field_width = int( win_width / cell_size )
# створюємо масив для клітин, він одномірний,
# але використовуючи допоміжну функцію addr –
# будемо перетворювати індекси 2-х мірний до 1-мірного
```

```
cell_matrix = []
тут у циклі створюємо екземпляри клітин
# і робимо їх прихованими
fori in range(field_height):
     forj in range(field_width):
               # тут створюємо екземпляр клітини # і
робимо його прихованими

          square = canvas.create_rectangle(2 + cell_size*j, 2 +
cell_size*i, cell_size + cell_size*j - 2, cell_size +
cell_size*i - 2, fill=fill_color)




          canvas.itemconfig(square, state=HIDDEN,
tags=('hid','0'))

           # «додаємо» до масиву
          cell_matrix.append(square)
# створимо фіктивний елемент, # він як би «повсюди»
поза полем

fict_square = canvas.create_rectangle(0,0,0,0, state=HIDDEN,
tags=('hid','0'))


cell_matrix.append(fict_square)
# задаємо «клітини» які знаходяться на полі # «за
замовчуванням»

canvas.itemconfig(cell_matrix[addr(8, 8)], state=NORMAL,
tags='vis')

canvas.itemconfig(cell_matrix[addr(10, 9)], state=NORMAL,
tags='vis')

canvas.itemconfig(cell_matrix[addr(9, 9)], state=NORMAL,
tags='vis')

canvas.itemconfig(cell_matrix[addr(9, 8)], state=NORMAL,
tags='vis')

canvas.itemconfig(cell_matrix[addr(9, 7)], state=NORMAL,
tags='vis')

canvas.itemconfig(cell_matrix[addr(10, 7)], state=NORMAL,
tags='vis')
```

```
# створюємо кадр для зберігання кнопок і # аналогічним чином, як з Canvas,

# встановлюємо кнопки дочірні кадри

frame = Frame(root)
# виконати «крок»
btn1 = Button (frame, text = 'Step', command = step) # Eval
# очистити поле
btn2 = Button (frame, text = 'Clear', command = clear)
# Зміна режимів «автоматично» - «по кроках»
btn3 = Button (frame, text = 'Run', command = run)
# пакуємо кнопки

btn1.pack(side='left')
btn2.pack(side='right')
btn3.pack(side='right')
# пакуємо кадр
frame.pack(side='bottom')

# прив'язуємо події кліка і руху миші над canvas до функції draw_a

canvas.bind('<B1-Motion>', draw_a)
canvas.bind('<ButtonPress>', draw_a)
через 500 мкс передати управління функції loop
root.after(500, loop)
стандартний цикл, що організовує події і загальну роботу віконного додатка

root.mainloop()
```

Різне

```
import sys
from math import *
з tkinter import *
з tkinter.filedialog import *
від tkinter.messagebox import *
#### http://younglinux.info/tkinter/canvas.php

deff1():
    print('f1 - run.........')
defb1(event):
     root.title("Ліва кнопка миші")
defb3(event):
     root.title("Права кнопка миші")
defmove(event):
     root.title("Рух мишею")
defres(event):
    root.title("зміна розміру вікна")
```

```
    print("зміна розміру вікна")
    xm, ym = xymax(root.geometry())
    print("--------xm---ym----------> ", xm, " ", ym)
    print("---------------------> ",canv. .geometry() )
    canv.config(width = xm, height = ym)

defxymax(str):
    """ отримаємо новий розмір вікна """
    print("===> ", str )
    k1=str.index('x')
    xmax1=str[:k1]
    k2=str.index('+',k1+1)
    ymax1=str[k1+1:k2]
    xmax = int (xmax1)
    ymax=int(ymax1)
    print(xmax,'',ymax)
    return xmax,ymax

defbutton_clicked():
    sss=root.geometry()
    print ("Клік!")
    print('root.maxsize()=<', root.maxsize(),'>' )
    print('canv.height()=<', canv.height(),'>' )
    print('canv.size()=<', canv.size(),'>' )
    print(root.geometry() )
    print('---canv.winfo_height()=',canv.winfo_height() )
     print('---canv.winfo_width() =',canv.winfo_width() )
     return
defwindow_deleted():
    print ('Вікно закрите')
    #canv.delete('all')
    if askyesno("Завершення програми", "Справді Ви хочете
завершити програму?"):
            #self.save_file()
            root.destroy()
    else:
            pass
   #root.destroy()
   # root.quit() # явна вказівка на вихід із програми
   # exit(0)
f = input('f(x):')

root = Tk()

f_1=Frame(root,bg="Yellow",)
f_2=Frame(root,bg="Blue")
f_1.pack(side=BOTTOM)
f_2.pack(side=BOTTOM)
button3 = Button(f_1, text=u"__B3 привіт", command=getF)
button3.pack(side=LEFT)

button4 = Button(f_2, text="_Б4 привіт", command=getF)
button4.pack(side=RIGHT) #LEFT)
```

```
canv = Canvas (root, width = 500, height = 500, bg = "white")

canv.create_line(500,1000,500,0,width=2,arrow=LAST)
canv.create_line(0,500,1000,500,width=2,arrow=LAST)

canv.create_line(200,50,300,50,width=3,fill="blue")
canv.create_line(0,0,100,100,width=2,arrow=LAST)

print('root.resizable()=<',root.resizable(),'>' )
print('root.root.maxsize()=<', root.maxsize(),'>' )

# кнопка за замовчуванням
button1 = Button(root, text=u"file read!", command=getF)
button1.pack(side=TOP) #LEFT) #'top') #place

# кнопка із зазначенням батьківського віджету та кількома
аргументами

button2 = Button(root, bg="red", text=u"Клікні мене!",
command=button_clicked)
button2.pack(side=TOP) #'top') #top')


root.protocol('WM_DELETE_WINDOW', window_deleted) # обробник
закриття вікна

#sys.exit(0)

First_x = -500;

forі in range(16000):
     if (i % 800 == 0):
          k = First_x + (1/16) * i
          canv.create_line(k + 500, -3 + 500, k + 500, 3 + 500,
width = 0.5, fill = 'black')
          canv.create_text(k + 515, -10 + 500, text = str(k),
                       fill="purple", font=("Helvectica", "10"))
          if (k! = 0):
               canv.create_line(-3 + 500, k + 500, 3 + 500,
                         k + 500, width = 0.5, fill = 'black')
               canv.create_text(20 + 500, k + 500, text = str(k),
                       fill="purple", font=("Helvectica", "10"))
     try:
          x = First_x + (1/16) * i
          new_f = f.replace('x', str(x))
          y = -eval(new_f) + 500
          x += 500
          # canv.create_oval(x, y, x + 1, y + 1, fill = 'black')
          canv.create_line(x, y, x + 1, y + 1, fill = 'black')
     except:
          pass
canv.pack()
```

```
################
root.bind('<Button-1>',b1)
root.bind('<Button-3>',b3)
root.bind('<Motion>',move)
root.bind('<Configure>',res)
#############
canv.focus_set()
root.mainloop()
print('@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@')
```