"""

Pygame base template for opening a window

Sample Python/Pygame Programs

Simpson College Computer Science

http://programarcadegames.com/

http://simpson.edu/computer-science/

Explanation video: http://youtu.be/vRB\_983kUMc

"""

import pygame

# Define some colors

BLACK = (0, 0, 0)

WHITE = (255, 255, 255)

GREEN = (0, 255, 0)

RED = (255, 0, 0)

BLUE = (0, 0, 255)

pygame.init()

# Set the width and height of the screen [width, height]

screen\_width = 650

screen\_height = 500

screen = pygame.display.set\_mode([screen\_width,screen\_height])

pygame.display.set\_caption("Instructions")

# Loop until the user clicks the close button.

done = False

# Used to manage how fast the screen updates

clock = pygame.time.Clock()

font = pygame.font.SysFont('Calibri', 25, True, False)

# -------- Main Program Loop -----------

while not done:

# --- Main event loop

for event in pygame.event.get():

if event.type == pygame.QUIT:

done = True

if event.type == pygame.MOUSEBUTTONDOWN:

import example as dm

# --- Game logic should go here

# --- Drawing code should go here

# First, clear the screen to white. Don't put other drawing commands

# above this, or they will be erased with this command.

screen.fill(BLUE)

text = font.render("""Welcome to my Game""" ,True,RED)

screen.blit(text, [100, 10])

text = font.render("""Control the chicken with the mouse""" ,True,RED)

screen.blit(text, [100, 40])

text = font.render("""Shoot with left mouse click""" ,True,RED)

screen.blit(text, [100, 70])

text = font.render("""Kill the hungry snakes before time runs out""" ,True,RED)

screen.blit(text, [100, 100])

text = font.render("""Click left mouse button to return to menu""" ,True,RED)

screen.blit(text, [100, 130])

# --- Go ahead and update the screen with what we've drawn.

pygame.display.flip()

# --- Limit to 60 frames per second

clock.tick(60)

# Close the window and quit.

# If you forget this line, the program will 'hang'

# on exit if running from IDLE.

pygame.quit()

"""

Pygame base template for opening a window

Sample Python/Pygame Programs

Simpson College Computer Science

http://programarcadegames.com/

http://simpson.edu/computer-science/

Explanation video: http://youtu.be/vRB\_983kUMc

"""

from tkinter import \*

import pygame

import random

import time

health=200

enemytimer=200

enemytimer1=0

time=300

arrows=[]

score = 0

# Define some colors

BLACK = (0, 0, 0)

WHITE = (255, 255, 255)

GREEN = (0, 255, 0)

RED = (255, 0, 0)

BLUE = (0, 0, 255)

player\_image = pygame.image.load("image/images/hunter.png")

fort = pygame.image.load("image/images/fort.png")

bullet = pygame.image.load("image/images/bullet.png")

enemy = pygame.image.load("image/images/enemy.png")

healthbar = pygame.image.load("image/images/healthbar.png")

health = pygame.image.load("image/images/health.png")

gameover = pygame.image.load("image/images/gameover.png")

grass = pygame.image.load("image/images/Grass.png")

gameover = pygame.image.load("image/images/gameover.png")

youwin = pygame.image.load("image/images/youwin.png")

#Intro

#Classes

class Enemy(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

my\_image=pygame.image.load("image/images/enemy.png")

self.image = my\_image

self.rect = self.image.get\_rect()

def update(self):

self.rect.x -= 3

if self.rect.x < 100:

self.rect.x = 670

class Player(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

my\_image=pygame.image.load("image/images/hunter.png")

self.image = my\_image

self.rect = self.image.get\_rect()

class Nest(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

my\_image=pygame.image.load("image/images/fort.png")

self.image = my\_image

self.rect = self.image.get\_rect()

class Bullet(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

my\_image=pygame.image.load("image/images/bullet.png")

self.image = my\_image

self.rect = self.image.get\_rect()

def update(self):

self.rect.x += 5

#Functions

def draw\_bullet(screen, x, y):

screen.blit(bullet,(x,y))

def draw\_hero(screen, x, y):

screen.blit(player\_image,(x,y))

def draw\_nests(screen, x, y):

screen.blit(fort,(0,140))

screen.blit(fort,(0,235))

screen.blit(fort,(0,350))

def draw\_enemies(screen, x, y):

screen.blit(enemy,(640,random.randint(50,430)))

# initialize pygame

pygame.init()

# Set the width and height of the screen [width, height]

screen\_width = 650

screen\_height = 500

screen = pygame.display.set\_mode([screen\_width,screen\_height])

pygame.display.set\_caption("My Game")

# Spitie lists

enemy\_list = pygame.sprite.Group()

# All Sprite List

all\_sprites\_list = pygame.sprite.Group()

# Bullet List

bullet\_list = pygame.sprite.Group()

# Loop to make the Spirites

for i in range(200):

# This represents a block

enemy = Enemy()

# Set a random location for the block

enemy.rect.x = random.randrange(600,650)

enemy.rect.y = random.randrange(screen\_height)

# Add the block to the list of objects

enemy\_list.add(enemy)

all\_sprites\_list.add(enemy)

#Creating the player

player = Player()

nest = Nest()

bullet = Bullet()

all\_sprites\_list.add(player,nest,bullet)

# Loop until the user clicks the close button.

done = False

# Used to manage how fast the screen updates

clock = pygame.time.Clock()

pygame.mouse.set\_visible(False)

#Create enemy and put them in list

#Add player to List

#all\_sprites\_list.add(player)

#create List for Bullets

#score thing

font = pygame.font.SysFont('Calibri', 25, True, False)

# -------- Main Program Loop -----------

while not done:

time -= 1

# --- Main event loop

for event in pygame.event.get():

if event.type == pygame.QUIT:

done = True

elif event.type == pygame.MOUSEBUTTONDOWN:

# Fire a bullet if the user clicks the mouse button

bullet = Bullet()

# Set the bullet so it is where the player is

bullet.rect.x = player.rect.x

bullet.rect.y = player.rect.y

# Add the bullet to the lists

all\_sprites\_list.add(bullet)

bullet\_list.add(bullet)

# --- Game logic should go here

# --- Drawing code should go here

# First, clear the screen to white. Don't put other drawing commands

# above this, or they will be erased with this command.

screen.fill(WHITE)

screen.blit(grass, [0, 0])

#Get dat mouse

pos = pygame.mouse.get\_pos()

#Got dat mouse

player.rect.x = pos[0]

player.rect.y = pos[1]

#moving the enemies + bullets

enemy\_list.update()

bullet\_list.update()

# sprite Collide Function

enemy\_hit\_list = pygame.sprite.spritecollide(bullet, enemy\_list, True)

#Check For Collisions

for enemy in enemy\_hit\_list:

score += 1

#Score thing

text = font.render("Score: " + str(score),True,BLUE)

screen.blit(text, [300, 10])

#time thing

text = font.render("Time Left: " + str(time),True,RED)

screen.blit(text, [100, 10])

# Draw all the spites

all\_sprites\_list.draw(screen)

#Checks for winner and loser

if score >= 200 and time >= 0:

screen.blit(youwin, (0,0))

if time <= 0:

screen.blit(gameover, (0,0))

# --- Go ahead and update the screen with what we've drawn.

pygame.display.flip()

# --- Limit to 60 frames per second

clock.tick(60)

# Close the window and quit.

# If you forget this line, the program will 'hang'

# on exit if running from IDLE.

pygame.quit()

"""

Pygame base template for opening a window

Sample Python/Pygame Programs

Simpson College Computer Science

http://programarcadegames.com/

http://simpson.edu/computer-science/

Explanation video: http://youtu.be/vRB\_983kUMc

"""

import pygame

import random

import time

health=200

enemytimer=200

enemytimer1=0

time=600

arrows=[]

score = 0

# Define some colors

BLACK = (0, 0, 0)

WHITE = (255, 255, 255)

GREEN = (0, 255, 0)

RED = (255, 0, 0)

BLUE = (0, 0, 255)

player\_image = pygame.image.load("image/images/hunter.png")

fort = pygame.image.load("image/images/fort.png")

bullet = pygame.image.load("image/images/bullet.png")

enemy = pygame.image.load("image/images/enemy.png")

healthbar = pygame.image.load("image/images/healthbar.png")

health = pygame.image.load("image/images/health.png")

gameover = pygame.image.load("image/images/gameover.png")

grass = pygame.image.load("image/images/Grass.png")

gameover = pygame.image.load("image/images/gameover.png")

youwin = pygame.image.load("image/images/youwin.png")

#Classes

class Enemy(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

my\_image=pygame.image.load("image/images/enemy.png")

self.image = my\_image

self.rect = self.image.get\_rect()

def update(self):

self.rect.x -= 3

if self.rect.x < 100:

self.rect.x = 670

class Player(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

my\_image=pygame.image.load("image/images/hunter.png")

self.image = my\_image

self.rect = self.image.get\_rect()

class Nest(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

my\_image=pygame.image.load("image/images/fort.png")

self.image = my\_image

self.rect = self.image.get\_rect()

class Bullet(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

my\_image=pygame.image.load("image/images/bullet.png")

self.image = my\_image

self.rect = self.image.get\_rect()

def update(self):

self.rect.x += 5

#Functions

def draw\_bullet(screen, x, y):

screen.blit(bullet,(x,y))

def draw\_hero(screen, x, y):

screen.blit(player\_image,(x,y))

def draw\_nests(screen, x, y):

screen.blit(fort,(0,140))

screen.blit(fort,(0,235))

screen.blit(fort,(0,350))

def draw\_enemies(screen, x, y):

screen.blit(enemy,(640,random.randint(50,430)))

# initialize pygame

pygame.init()

# Set the width and height of the screen [width, height]

screen\_width = 650

screen\_height = 500

screen = pygame.display.set\_mode([screen\_width,screen\_height])

pygame.display.set\_caption("My Game")

# Spitie lists

enemy\_list = pygame.sprite.Group()

# All Sprite List

all\_sprites\_list = pygame.sprite.Group()

# Bullet List

bullet\_list = pygame.sprite.Group()

# Loop to make the Spirites

for i in range(10):

# This represents a block

enemy = Enemy()

# Set a random location for the block

enemy.rect.x = random.randrange(600,650)

enemy.rect.y = random.randrange(screen\_height)

# Add the block to the list of objects

enemy\_list.add(enemy)

all\_sprites\_list.add(enemy)

#Creating the player

player = Player()

nest = Nest()

bullet = Bullet()

all\_sprites\_list.add(player,nest,bullet)

# Loop until the user clicks the close button.

done = False

# Used to manage how fast the screen updates

clock = pygame.time.Clock()

pygame.mouse.set\_visible(False)

#Create enemy and put them in list

#Add player to List

#all\_sprites\_list.add(player)

#create List for Bullets

#score thing

font = pygame.font.SysFont('Calibri', 25, True, False)

# -------- Main Program Loop -----------

while not done:

time -= 1

# --- Main event loop

for event in pygame.event.get():

if event.type == pygame.QUIT:

done = True

elif event.type == pygame.MOUSEBUTTONDOWN:

# Fire a bullet if the user clicks the mouse button

bullet = Bullet()

# Set the bullet so it is where the player is

bullet.rect.x = player.rect.x

bullet.rect.y = player.rect.y

# Add the bullet to the lists

all\_sprites\_list.add(bullet)

bullet\_list.add(bullet)

# --- Game logic should go here

# --- Drawing code should go here

# First, clear the screen to white. Don't put other drawing commands

# above this, or they will be erased with this command.

screen.fill(WHITE)

screen.blit(grass, [0, 0])

#Get dat mouse

pos = pygame.mouse.get\_pos()

#Got dat mouse

player.rect.x = pos[0]

player.rect.y = pos[1]

#moving the enemies + bullets

enemy\_list.update()

bullet\_list.update()

# sprite Collide Function

enemy\_hit\_list = pygame.sprite.spritecollide(bullet, enemy\_list, True)

#Check For Collisions

for enemy in enemy\_hit\_list:

score += 1

#Score thing

text = font.render("Score: " + str(score),True,BLUE)

screen.blit(text, [300, 10])

#time thing

text = font.render("Time Left: " + str(time),True,RED)

screen.blit(text, [100, 10])

# Draw all the spites

all\_sprites\_list.draw(screen)

#Checks for winner and loser

if score >= 10 and time >= 0:

screen.blit(youwin, (0,0))

if time <= 0:

screen.blit(gameover, (0,0))

# --- Go ahead and update the screen with what we've drawn.

pygame.display.flip()

# --- Limit to 60 frames per second

clock.tick(60)

# Close the window and quit.

# If you forget this line, the program will 'hang'

# on exit if running from IDLE.

pygame.quit()

import pygame

import dumbmenu as dm

pygame.init()

# Just a few static variables

red = 255, 0, 0

green = 0,255, 0

blue = 0, 0,255

size = width, height = 650,500

screen = pygame.display.set\_mode(size)

screen.fill(blue)

pygame.display.update()

pygame.key.set\_repeat(500,30)

choose = dm.dumbmenu(screen, [

'Start Game',

'Credits',

'Instructions',

'Change Difficulty to Impossible',

'Quit Game'], 64,64,None,32,1.4,green,red)

if choose == 0:

import Gamecompleteeasy as dm

elif choose == 1:

import credits as dm

elif choose == 2:

import instructions as dm

elif choose == 3:

import Gamecompleteimpossible as dm

elif choose == 4:

print ("You choose 'Quit Game'.")

pygame.quit()

exit()

#!/usr/bin/env python

# -\*- coding: utf-8 -\*-

import pygame, sys

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Function "dumbmenu" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def dumbmenu(screen, menu, x\_pos = 100, y\_pos = 100, font = None,

size = 70, distance = 1.4, fgcolor = (255,255,255),

cursorcolor = (255,0,0), exitAllowed = True):

"""Draws a Menu using pygame.

Parameters are: screen, menu, x\_pos, y\_pos, font,

size, distance, fgcolor, cursor

PARAMETERS

==========

screen (Surface): The surface created with pygame.display.set\_mode()

menu (List): A List of every menupoint that should be visible

x\_pos (digit): Start of x\_position, in Pixels (Default: 100)

y\_pos (digit): Start of y\_position, in Pixels (Default: 100)

size (digit): Fontsize (Default: 70)

distance (float):Y-Distance after every single menupoint

(Default: 1.4)

fgcolor (Tupel): Foreground-Color, means the Font-Color

(Default: (255,255,255), means white)

cursorcolor (Tupel): Cursor-Color, means that ">"-Charakter

(Default: (255,0,0), means red)

exitAllowed (Bool): If True:

If User pressed the ESC-Key, the Cursor will

move to the last Menupoint. If Cursorposition

is already to the last Menupoint, a pressed

ESC-Key will return the latest Menupoint. Very

useful if the last Menupoint is something like

"Quit Game"...

If False:

A pressed ESC-Key will takes no effect.

(Default: True)

EXAMPLE

=======

import pygame

from dumbmenu import \*

pygame.init()

# Just a few static variables

red = 255, 0, 0

green = 0,255, 0

size = width, height = 640,480

screen = pygame.display.set\_mode(size)

screen.fill(blue)

pygame.display.update()

print dumbmenu(screen, [

'Start Game',

'Options',

'Manual',

'Show Highscore',

'Quit Game'],

320, 250, "Courier", 32, 1.4, green, red)

HOW TO INTERACT

===============

After called dumbmenu(), the User MUST choose an Menupoint. The

Script will be haltet until the User makes a decision or a event

called pygame.QUIT() will be raised.

The User kann pressed directly a Key from 1 to 9 to take the choice.

Another Method is pressing the UP-/DOWN-Key and take the choice with

RETURN. Every single Menupoint will get a Number, beginning with 1.

The return-value ist the Number of Menupoint decreased by 1. From

the above Example: If the User will choice "Manual", the return-

value will be 2.

If the number of Menupoints is greater than 9, the numeration will

continue from A to Z... the return-value is still a number,

continue from 9 to 34...

If a pygame.QUIT()-Event will be raised, the return-value will be

-1.

ACTUAL LIMITATIONS

==================

It's actually not possible to change the Font itself.

Drawing Menu will be antialiased. If you want to change that, you'll

have to change the sourcecode directly.

OTHERS

======

Yes, I know, my english isn't that good (I'm not a naturally

speaker) and the sourcecode isn't that good too ;) . It's more or

less a "quick'n dirty"-Solution. My first intention was to make that

code for me, but I hope it could may useful for other people too...

Version: 0.40

Author: Manuel Kammermeier aka Astorek

License: MIT

CHANGES:

========

Version 0.35:

- First Version

Version 0.40:

- "bgcolor" removed, now the Function saves the Background

- added "font", which allows to choose a Systemfont

"""

# Draw the Menupoints

pygame.font.init()

if font == None:

myfont = pygame.font.Font(None, size)

else:

myfont = pygame.font.SysFont(font, size)

cursorpos = 0

renderWithChars = False

for i in menu:

if renderWithChars == False:

text = myfont.render(str(cursorpos + 1)+". " + i,

True, fgcolor)

else:

text = myfont.render(chr(char)+". " + i,

True, fgcolor)

char += 1

textrect = text.get\_rect()

textrect = textrect.move(x\_pos,

(size // distance \* cursorpos) + y\_pos)

screen.blit(text, textrect)

pygame.display.update(textrect)

cursorpos += 1

if cursorpos == 9:

renderWithChars = True

char = 65

# Draw the ">", the Cursor

cursorpos = 0

cursor = myfont.render(">", True, cursorcolor)

cursorrect = cursor.get\_rect()

cursorrect = cursorrect.move(x\_pos - (size // distance),

(size // distance \* cursorpos) + y\_pos)

# The whole While-loop takes care to show the Cursor, move the

# Cursor and getting the Keys (1-9 and A-Z) to work...

ArrowPressed = True

exitMenu = False

clock = pygame.time.Clock()

filler = pygame.Surface.copy(screen)

fillerrect = filler.get\_rect()

while True:

clock.tick(30)

if ArrowPressed == True:

screen.blit(filler, fillerrect)

pygame.display.update(cursorrect)

cursorrect = cursor.get\_rect()

cursorrect = cursorrect.move(x\_pos - (size // distance),

(size // distance \* cursorpos) + y\_pos)

screen.blit(cursor, cursorrect)

pygame.display.update(cursorrect)

ArrowPressed = False

if exitMenu == True:

break

for event in pygame.event.get():

if event.type == pygame.QUIT:

return -1

if event.type == pygame.KEYDOWN:

if event.key == pygame.K\_ESCAPE and exitAllowed == True:

if cursorpos == len(menu) - 1:

exitMenu = True

else:

cursorpos = len(menu) - 1; ArrowPressed = True

# This Section is huge and ugly, I know... But I don't

# know a better method for this^^

if event.key == pygame.K\_1:

cursorpos = 0; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_2 and len(menu) >= 2:

cursorpos = 1; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_3 and len(menu) >= 3:

cursorpos = 2; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_4 and len(menu) >= 4:

cursorpos = 3; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_5 and len(menu) >= 5:

cursorpos = 4; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_6 and len(menu) >= 6:

cursorpos = 5; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_7 and len(menu) >= 7:

cursorpos = 6; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_8 and len(menu) >= 8:

cursorpos = 7; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_9 and len(menu) >= 9:

cursorpos = 8; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_a and len(menu) >= 10:

cursorpos = 9; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_b and len(menu) >= 11:

cursorpos = 10; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_c and len(menu) >= 12:

cursorpos = 11; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_d and len(menu) >= 13:

cursorpos = 12; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_e and len(menu) >= 14:

cursorpos = 13; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_f and len(menu) >= 15:

cursorpos = 14; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_g and len(menu) >= 16:

cursorpos = 15; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_h and len(menu) >= 17:

cursorpos = 16; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_i and len(menu) >= 18:

cursorpos = 17; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_j and len(menu) >= 19:

cursorpos = 18; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_k and len(menu) >= 20:

cursorpos = 19; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_l and len(menu) >= 21:

cursorpos = 20; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_m and len(menu) >= 22:

cursorpos = 21; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_n and len(menu) >= 23:

cursorpos = 22; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_o and len(menu) >= 24:

cursorpos = 23; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_p and len(menu) >= 25:

cursorpos = 24; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_q and len(menu) >= 26:

cursorpos = 25; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_r and len(menu) >= 27:

cursorpos = 26; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_s and len(menu) >= 28:

cursorpos = 27; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_t and len(menu) >= 29:

cursorpos = 28; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_u and len(menu) >= 30:

cursorpos = 29; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_v and len(menu) >= 31:

cursorpos = 30; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_w and len(menu) >= 32:

cursorpos = 31; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_x and len(menu) >= 33:

cursorpos = 32; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_y and len(menu) >= 34:

cursorpos = 33; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_z and len(menu) >= 35:

cursorpos = 34; ArrowPressed = True; exitMenu = True

elif event.key == pygame.K\_UP:

ArrowPressed = True

if cursorpos == 0:

cursorpos = len(menu) - 1

else:

cursorpos -= 1

elif event.key == pygame.K\_DOWN:

ArrowPressed = True

if cursorpos == len(menu) - 1:

cursorpos = 0

else:

cursorpos += 1

elif event.key == pygame.K\_KP\_ENTER or \

event.key == pygame.K\_RETURN:

exitMenu = True

return cursorpos

if \_\_name\_\_ == '\_\_main\_\_':

sys.stderr.write("You should import me, not start me...")

sys.exit()

"""

Pygame base template for opening a window

Sample Python/Pygame Programs

Simpson College Computer Science

http://programarcadegames.com/

http://simpson.edu/computer-science/

Explanation video: http://youtu.be/vRB\_983kUMc

"""

import pygame

# Define some colors

BLACK = (0, 0, 0)

WHITE = (255, 255, 255)

GREEN = (0, 255, 0)

RED = (255, 0, 0)

BLUE = (0, 0, 255)

pygame.init()

# Set the width and height of the screen [width, height]

screen\_width = 650

screen\_height = 500

screen = pygame.display.set\_mode([screen\_width,screen\_height])

pygame.display.set\_caption("Credits")

# Loop until the user clicks the close button.

done = False

# Used to manage how fast the screen updates

clock = pygame.time.Clock()

font = pygame.font.SysFont('Calibri', 25, True, False)

# -------- Main Program Loop -----------

while not done:

# --- Main event loop

for event in pygame.event.get():

if event.type == pygame.QUIT:

done = True

if event.type == pygame.MOUSEBUTTONDOWN:

import example as dm

# --- Game logic should go here

# --- Drawing code should go here

# First, clear the screen to white. Don't put other drawing commands

# above this, or they will be erased with this command.

screen.fill(BLUE)

text = font.render("""Credits""" ,True,RED)

screen.blit(text, [100, 10])

text = font.render("""Made by: Marcos Vazquez and Gian Carlo""" ,True,RED)

screen.blit(text, [100, 40])

text = font.render("""Graphics Design Marcos Vazquez""" ,True,RED)

screen.blit(text, [100, 70])

text = font.render("""Coding done by Marcos Vazquez and Gian Carlo """ ,True,RED)

screen.blit(text, [100, 100])

text = font.render("""Click left mouse button to return to menu""" ,True,RED)

screen.blit(text, [100, 130])

# --- Go ahead and update the screen with what we've drawn.

pygame.display.flip()

# --- Limit to 60 frames per second

clock.tick(60)

# Close the window and quit.

# If you forget this line, the program will 'hang'

# on exit if running from IDLE.

pygame.quit()