```
# By Alberto Caro S.
# Ing. Civil en Computacion
# Doctor(c) Cs. de la Computacion
# Pontificia Universidad Catolica de Chile
# Programacion de Robot -> INFO1139
#------------------------------------------------------------------------
#__________      ___.           __  .__
#\______   \ ____\_ |__   _____/  |_|__| ____ _____
# |       _//  _ \| __ \ /  _ \   __\  |/ ___\\__  \
# |    |   (  <_> ) \_\ (  <_> )  | |  \  \___ /  __ \_
# |____|_  /\____/|___  /\____/|__| |__|\___  >____  /
#        \/           \/                    \/     \/
#------------------------------------------------------------------------
import pygame,time, random as RA, ctypes as ct
from pygame.locals import *

#------------------------------------------------------------------------
# Definicion de Constantes Globales
#------------------------------------------------------------------------
nRES = (1200,700) ; nMAX_ROBOTS  = 100 ;nMAX_NAVES = 2  ; xd = 0
nMIN_X = 0 ; nMAX_X = 2640 ; nMIN_Y = 0 ; nMAX_Y = 1760 ; yd = 0
nTRUE  = 1 ; nTIME1 = 400 ; nTIME2 = 400 ; lOK = True
nTILE_WX = 44 ; nTILE_HY = 44 ; nX0 = 232 ; nY0 = 14 ; nBTN_LEFT = 1

#------------------------------------------------------------------------
# Definicion de Estructura de Datos.-
#
# nF : Identifica al Robot (Sprite)
# nX : Coordenada X del Robot
# nY : Coordenada Y del Robot
# nR : Rango a mover del robot -> Cuantos pixeles intentara moverse
# dX : Direccion en Eje X ->
#                            0 : No se mueve
#                           +1 : Se Mueve un pixel a la Derecha
#                           -1 : Se Mueve un pixel a la Izquierda
# dY :
#                            0 : No se mueve
#                           +1 : Se Mueve un pixel hacia Abajo
#                           -1 : Se Mueve un pixel hacia Arriba
# nV : Velocidad del robot -> (0,1,2,3, ...etc) Aleatoria

#------------------------------------------------------------------------
class eRobot(ct.Structure):
 _fields_ = [
             ('nF',ct.c_short),('nX',ct.c_short),('nY',ct.c_short),
             ('nR',ct.c_short),('dX',ct.c_short),('dY',ct.c_short),
             ('nV',ct.c_short)
            ]

#------------------------------------------------------------------------
# Definicion de Estructura de Datos.-
#
# Identica a la estructura de los robots, pero es para las Naves.-
#------------------------------------------------------------------------
class eNaves(ct.Structure):
 _fields_ = [
             ('nF',ct.c_short),('nX',ct.c_short),('nY',ct.c_short),
             ('nR',ct.c_short),('dX',ct.c_short),('dY',ct.c_short),
             ('nV',ct.c_short)
            ]

#------------------------------------------------------------------------
# Carga imagenes y convierte formato PyGame
#------------------------------------------------------------------------
def Load_Image(sFile,transp = False):
    try: image = pygame.image.load(sFile)
    except pygame.error,message:
           raise SystemExit,message
    image = image.convert()
    if transp:
       color = image.get_at((0,0))
       image.set_colorkey(color,RLEACCEL)
    return image
```

```
74
75  #--------------------------------------------------------------------
76  # Inicializa PyGames.-
77  # Inicializa el Engine de PyGame
78  # Define 2 Timers (contadores/relojes) a los cuales se les puede
79  # asignar cualquier SCRIPTS a ejecutar de manera independiente de
80  # la logica o secuencia del SCRIPT principal
81  #--------------------------------------------------------------------
82  def PyGame_Init():
83      pygame.init()
84      pygame.time.set_timer(USEREVENT+1,nTIME1)
85      pygame.time.set_timer(USEREVENT+2,nTIME2)
86      pygame.mouse.set_visible(False) # Hacemos invisible el cursos del Mouse
87      pygame.display.set_caption('Demo Robot Mapa2D - By Alberto Caro')
88      return pygame.display.set_mode(nRES) # Retornamos la Surface principal
89                                           # de 1.200 x 700 RGB
90
91  #--------------------------------------------------------------------
92  # Inicializa las Baldozas = Tiles del Super Extra Mega Mapa.-
93  # Las Baldosas = Tiles miden : X -> 44 px, Y -> 44 px
94  # Se crea y llena de manera aleatoria un SUPER Mapa de 2.640x1.760 px
95  # Asi en:
96  #            2.640 / 44 = 60 Tiles en Eje X
97  #            1.760 / 44 = 40 Tiles en Eje Y
98  #
99  #         << nTILE_WX = 44 , nTILE_HY = 44 >>
100 # Se retorna un Array de 2D con 60 Columnas y 40 Filas con valores
101 # Aleatorios entre [0,1,2,3,4,5,6,7,8,9]
102 #--------------------------------------------------------------------
103 def Tiles_Init():
104     return [[RA.randint(0,9) for i in range(0,nMAX_X/nTILE_WX)] \
105                              for i in range(0,nMAX_Y/nTILE_HY)]
106
107 #--------------------------------------------------------------------
108 # Inicializa Superficie del Super Extra Mega Mapa.-
109 # Se crea y retorna el Super Mapa de 2.640 x 1.760 px
110 # Es decir, retorna una Surface de este tamano donde se almacenara el
111 # Mapa mediante los Tiles/Baldosas
112 #--------------------------------------------------------------------
113 def Mapa_Init(nAncho_X,nAlto_Y):
114     return pygame.Surface((nAncho_X,nAlto_Y))
115
116 #--------------------------------------------------------------------
117 # Inicializa Array de Sprites.-
118 # aImg es un Array donde sus celdas contienen las Baldosas (0...9)
119 # Los 3 Tipos de Robots (0,1,2) y las 2 Naves (0,1)
120 # Panel Main (Bkg), 2 Cursores y un MiniMapa
121 # Retorna el contenido de las Superficies anteriores dentro del Array
122 #--------------------------------------------------------------------
123 def Fig_Init():
124     aImg = []
125     aImg.append(Load_Image('f0.png',False )) # Baldosa 0               0
126     aImg.append(Load_Image('f1.png',False )) # Baldosa 1               1
127     aImg.append(Load_Image('f2.png',False )) # Baldosa 2               2
128     aImg.append(Load_Image('f3.png',False )) # Baldosa 3               3
129     aImg.append(Load_Image('f4.png',False )) # Baldosa 4               4
130     aImg.append(Load_Image('f5.png',True  )) # Baldosa 5               5
131     aImg.append(Load_Image('f6.png',True  )) # Baldosa 6               6
132     aImg.append(Load_Image('f7.png',True  )) # Baldosa 7               7
133     aImg.append(Load_Image('f8.png',True  )) # Baldosa 8               8
134     aImg.append(Load_Image('f9.png',True  )) # Baldosa 9               9
135     aImg.append(Load_Image('fa.png',True  )) # Nave 1                 10
136     aImg.append(Load_Image('fb.png',True  )) # Nave 2                 11
137     aImg.append(Load_Image('fc.png',True  )) # Robot 1 Led R-V        12
138     aImg.append(Load_Image('fd.png',True  )) # Robot 2 Led V-R        13
139     aImg.append(Load_Image('fe.png',True  )) # Robot 3 Timer          14
140     aImg.append(Load_Image('ff.png',True  )) # Cursor Mouse 1 Normal  15
141     aImg.append(Load_Image('pm.png',True  )) # Cursor Mouse 2 Mini    16
142     aImg.append(Load_Image('bg.png',True  )) # Panel Main             17
143     aImg.append(Load_Image('mm.png',False )) # Mini Mapa              18
144     return aImg
145
146 #--------------------------------------------------------------------
```

```
147  # Inicilaiza parametros de los Robots
148  #---------------------------------------------------------------------
149  def Robots_Init():
150      for i in range(0,nMAX_ROBOTS):
151       aBoes[i].nF = RA.randint(1,3)    # Identifica al Robot
152       aBoes[i].nX = RA.randint(0,2639) # nMAX_X-nTILE_WX)  # Pos. X Robot Mapa
153       aBoes[i].nY = RA.randint(0,1759) # nMAX_Y-nTILE_HY)  # Pos. Y Robot Mapa
154       aBoes[i].nR = RA.randint(0,500)  # Rango de Desplazamiento.-
155       aBoes[i].dX = 0 # Sin movimiento por eje X
156       aBoes[i].dY = 0 # Sin movimiento por eje Y
157       aBoes[i].nV = RA.randint(1,3) # Velocidad Aleatoria entre [1,2,3]
158      return
159
160  #---------------------------------------------------------------------
161  # Incializa parametros de las Naves.-
162  #---------------------------------------------------------------------
163  def Naves_Init():
164      for i in range(0,nMAX_NAVES):
165       aNave[i].nF = i   # Identifica a la Nave
166       aNave[i].nX = RA.randint(100,200) # Pos. X Robot Mapa
167       aNave[i].nY = 600 # Pos. Y Robot Mapa
168       aNave[i].nR = RA.randint(0,100)# Rango de Desplazamiento.-
169       aNave[i].dX = 0  # Sin movimiento por eje X
170       aNave[i].dY = -1 # Se movera hacia Arriba al inicio
171       aNave[i].nV = RA.randint(1,2) # Velocidad entre [1,2]
172      return
173
174  #---------------------------------------------------------------------
175  # Pinta los Robots en el Super Extra Mega Mapa.-
176  # Se pintan los Robots en Surface -> sMapa (2.640 x 1.760)
177  #---------------------------------------------------------------------
178  def Pinta_Robots():
179      for i in range(0,nMAX_ROBOTS):
180       if aBoes[i].nF == 1: #  Robot tipo 1?
181          sMapa.blit(aSprt[12],(aBoes[i].nX,aBoes[i].nY))
182       if aBoes[i].nF == 2: #  Robot tipo 2?
183          sMapa.blit(aSprt[13],(aBoes[i].nX,aBoes[i].nY))
184       if aBoes[i].nF == 3: #  Robot tipo 3?
185          sMapa.blit(aSprt[14],(aBoes[i].nX,aBoes[i].nY))
186      return
187
188  #---------------------------------------------------------------------
189  # Pinta las Naves en el Panel Izquierdo
190  # Utiliza la Surface -> Panta (Principal) 1.200 x 700 px
191  #---------------------------------------------------------------------
192  def Pinta_Naves():
193      Panta.blit(aSprt[10],(aNave[0].nX,aNave[0].nY))
194      Panta.blit(aSprt[11],(aNave[1].nX,aNave[1].nY))
195      return
196
197  #---------------------------------------------------------------------
198  # Pinta la Pantalla Principal de PyGames.-
199  # Pinta el BakGround en Panta(Surface tipo Display Main)
200  #---------------------------------------------------------------------
201  def Pinta_Panel():
202      Panta.blit(aSprt[17],(0,0))
203      return
204
205  #---------------------------------------------------------------------
206  # Pinta el Super Extra Mega Mapa a Panta de PyGames.-
207  # Saca una copia del Mapa Ppal sMapa del tamano de 952 px de Ancho
208  # por 670 px de Alto, a partir de las coordenadas de inicio de
209  # (nX0,nY0) las cuales pueden ir variando de acuerdo a la posicion del
210  # Cursor del Mouse.
211  #---------------------------------------------------------------------
212  def Pinta_Mapa():
213      Panta.blit(sMapa.subsurface((xd,yd,952,670)),(nX0,nY0))
214      return
215
216  #---------------------------------------------------------------------
217  # Pinta el Mini Mapa a Panta de PyGames.-
218  # Sobre este MiniMapa que se superpone a Panta, se pintan los robots
219  # de manera simbolica (unos cuadraditos pequenos).
```

```
220   # Para ello se realiza una InterPolacin (Escala) del Mini Mapa al Mapa
221   # Grande
222   #---------------------------------------------------------------------
223   def Pinta_MMapa():
224       xp = 0; yp = 0
225       Panta.blit(aSprt[18],(1013,20))
226       for i in range(0,nMAX_ROBOTS):
227        xp = int(159/float(2640)*aBoes[i].nX) + 1017
228        xy = int(112/float(1760)*aBoes[i].nY) + 0027
229        Panta.blit(aSprt[16],(xp,xy))
230       return
231
232   #---------------------------------------------------------------------
233   # Pinta la Posicion de la Mouse en Panta de PyGame.-
234   #---------------------------------------------------------------------
235   def Pinta_Mouse():
236       Panta.blit(aSprt[15],(nMx,nMy))
237       return
238
239   #---------------------------------------------------------------------
240   # Acualiza Coordenadas Scroll Super Extra Mega Mapa.-
241   # Se realiza un Scroll del Mapa segun las coordenadas del mouse pero
242   # dentro del Mini Mapa.-
243   #---------------------------------------------------------------------
244   def UpDate_Scroll_Mapa(nMx,nMy):
245       xd = 0 ; yd = 0
246       if nMx in range(1018,1177):
247          if nMy in range(25,137):
248             xd = int(2640*(nMx-1018)/float(159))
249             yd = int(1760*(nMy-25)/float(112))
250             pygame.display.set_caption('[Coord Mapa]-> X: %d - Y: %d' %(xd,yd))
251             if xd >= 1687: xd = 1687
252             if yd >= 1090: yd = 1090
253       return xd,yd
254
255   #---------------------------------------------------------------------
256   # Acualiza el Super Extra Mega Mapa.-
257   # Pinta el Mapa Completo de sMapa segun los tiles asociados en el array.
258   #---------------------------------------------------------------------
259   def UpDate_Mapa():
260       nPx = nPy = 0
261       for f in range(0,nMAX_Y/nTILE_HY):
262        for c in range(0,nMAX_X/nTILE_WX):
263         if aTile[f][c] == 0: # Tile 0
264            sMapa.blit(aSprt[0],(nPx,nPy)); nPx += nTILE_WX
265         if aTile[f][c] == 1: # Tile 1
266            sMapa.blit(aSprt[1],(nPx,nPy)); nPx += nTILE_WX
267         if aTile[f][c] == 2: # Tile 2
268            sMapa.blit(aSprt[1],(nPx,nPy)); nPx += nTILE_WX
269         if aTile[f][c] == 3: # Tile 3
270            sMapa.blit(aSprt[3],(nPx,nPy)); nPx += nTILE_WX
271         if aTile[f][c] == 4: # Tile 4
272            sMapa.blit(aSprt[4],(nPx,nPy)); nPx += nTILE_WX
273         if aTile[f][c] == 5: # Tile 5
274            sMapa.blit(aSprt[1],(nPx,nPy)); nPx += nTILE_WX
275         if aTile[f][c] == 6: # Tile 6
276            sMapa.blit(aSprt[1],(nPx,nPy)); nPx += nTILE_WX
277         if aTile[f][c] == 7: # Tile 7
278            sMapa.blit(aSprt[1],(nPx,nPy)); nPx += nTILE_WX
279         if aTile[f][c] == 8: # Tile 8
280            sMapa.blit(aSprt[1],(nPx,nPy)); nPx += nTILE_WX
281         if aTile[f][c] == 9: # Tile 9
282            sMapa.blit(aSprt[4],(nPx,nPy)); nPx += nTILE_WX
283        nPx = 0; nPy += nTILE_HY
284       return
285
286   #---------------------------------------------------------------------
287   # Actualiza la estructura de datos de cada uno de los robots dentro del
288   # Mapa sMapa.
289   #---------------------------------------------------------------------
290   def UpDate_Robots():
291       for i in range(0,nMAX_ROBOTS): # Recorrimos todos los Robots
292        aBoes[i].nR -= 1    # Decrementamos en 1 el Rango del Robot
```

```
293         if aBoes[i].nR < 0: # Si es negativo ->
294            aBoes[i].nR = RA.randint(0,500) # Asignamos otro Rango aleatorio
295            aBoes[i].nV = RA.randint(1,3)   # Asignamos otra velocidad
296            nDir = RA.randint(1,9)  # Generamos una Direccion de Movimiento Aleat.
297            if nDir == 1: # Norte ?
298               aBoes[i].dX = +0 ; aBoes[i].dY = -1
299            if nDir == 2: # Este ?
300               aBoes[i].dX = +1 ; aBoes[i].dY = 0
301            if nDir == 3: # Sur ?
302               aBoes[i].dX = +0 ; aBoes[i].dY = +1
303            if nDir == 4: # Oeste ?
304               aBoes[i].dX = -1 ; aBoes[i].dY = +0
305            if nDir == 5: # Detenido ?
306               aBoes[i].dX = +0 ; aBoes[i].dY = +0
307            if nDir == 6: # NorEste
308               aBoes[i].dX = +1 ; aBoes[i].dY = -1
309            if nDir == 7: # NorWeste
310               aBoes[i].dX = -1 ; aBoes[i].dY = -1
311            if nDir == 8: # SurEste
312               aBoes[i].dX = +1 ; aBoes[i].dY = +1
313            if nDir == 9: # SurWeste
314               aBoes[i].dX = -1 ; aBoes[i].dY = +1
315
316         #Actualizamos (Xs,Ys) de los Sprites en el Mapa 2D
317         #-------------------------------------------------
318
319         aBoes[i].nX += aBoes[i].dX*aBoes[i].nV # Posicion Robot[i] en eje X
320         aBoes[i].nY += aBoes[i].dY*aBoes[i].nV # Posicion Robot[i] en eje Y
321
322         if aBoes[i].nX < nMIN_X: # Check los bordes o limites
323            aBoes[i].nX = nMIN_X ; aBoes[i].nR = 0 # Flag
324
325         if aBoes[i].nX > (nMAX_X - nTILE_WX): # Check los bordes o limites
326            aBoes[i].nX = nMAX_X - nTILE_WX ; aBoes[i].nR = 0 # Flag
327
328         if aBoes[i].nY < nMIN_Y: # Check los bordes o limites
329            aBoes[i].nY = nMIN_Y ; aBoes[i].nR = 0 # Flag
330
331         if aBoes[i].nY > (nMAX_Y - nTILE_HY): # Check los bordes o limites
332            aBoes[i].nY = nMAX_Y - nTILE_HY ; aBoes[i].nR = 0 # Flag
333
334       return
335
336   #-----------------------------------------------------------------------
337   # Actualiza las Naves en el Panel Izquierdo.-
338   #-----------------------------------------------------------------------
339   def UpDate_Naves():
340       for i in range(0,nMAX_NAVES):
341        aNave[i].nR -= 1
342        if aNave[i].nR < 0:
343           aNave[i].nR = RA.randint(0,100)
344           aNave[i].nV = RA.randint(1,2)
345           nDir = RA.randint(1,5)
346           if nDir == 1: # Norte ?
347              aNave[i].dX = +0 ; aNave[i].dY = -1
348           if nDir == 2: # Este ?
349              aNave[i].dX = +1 ; aNave[i].dY = 0
350           if nDir == 3: # Sur ?
351              aNave[i].dX = +0 ; aNave[i].dY = +1
352           if nDir == 4: # Oeste ?
353              aNave[i].dX = -1 ; aNave[i].dY = +0
354           if nDir == 5: # Detenido ?
355              aNave[i].dX = +0 ; aNave[i].dY = +0
356
357        #Actualizamos (Xs,Ys) de los Sprites en el Mapa 2D
358        #-------------------------------------------------
359
360        aNave[i].nX += aNave[i].dX*aNave[i].nV # Posicion en Eje X de Nave[i]
361        aNave[i].nY += aNave[i].dY*aNave[i].nV # Posicion en Eje X de Nave[i]
362
363        # Check bordes o limites de las naves en su sector
364        if aNave[i].nX < 017 : aNave[i].nX = 017 ; aNave[i].nR = 0 # Flag
365        if aNave[i].nX > 156 : aNave[i].nX = 156 ; aNave[i].nR = 0
```

```
366         if aNave[i].nY < 052 : aNave[i].nY = 052 ; aNave[i].nR = 0
367         if aNave[i].nY > 600 : aNave[i].nY = 600 ; aNave[i].nR = 0
368 
369       return
370 
371   #-----------------------------------------------------------------
372   # Handle de Pause.-
373   # Pausamos la ejecucion de la aplicacion
374   #-----------------------------------------------------------------
375   def Pausa():
376       while 1:
377        e = pygame.event.wait()
378        if e.type in (pygame.QUIT, pygame.KEYDOWN):
379           return
380 
381   #-----------------------------------------------------------------------
382   # While Principal del Demo.-
383   #-----------------------------------------------------------------------
384   Panta = PyGame_Init() # Surface tipo Surface Display principal
385   aSprt = Fig_Init()    # Cargamos los graficos al Array aSprt
386   aTile = Tiles_Init(); # Inicializamos el Array de Tiles
387   sMapa = Mapa_Init(2640,1760) # Creamos el Super Mapa -> sMapa (Surface)
388   aBoes = [ eRobot() for i in range(0,nMAX_ROBOTS) ] # Array de Robots
389   aNave = [ eNaves() for i in range(0,nMAX_NAVES)  ] # Array de Naves
390   Clok  = pygame.time.Clock(); nMx = 0; nMy = 0 # Clock para Syncronize
391 
392   Robots_Init() # Inicializamos todas las Estructuras de Datos de los Robots
393   Naves_Init()  # Inicializamos todas las Estructuras de Datos de las Naves
394 
395   # While principal y logica de llamadas y salida del programa
396   while lOK:
397    cKey = pygame.key.get_pressed() # Se presiono alguna tecla?
398    if cKey[pygame.K_ESCAPE] : lOK = False
399    if cKey[pygame.K_p]      : Pausa() # Tecla 'P' -> Pausa
400    if cKey[pygame.K_s]      : pygame.image.save(Panta,'Capture.png') # Captura
401    ev = pygame.event.get()
402    for e in ev:
403     if e.type == QUIT               : lOK = False
404     if e.type == pygame.MOUSEMOTION : nMx,nMy = e.pos # Coordenada Mouse
405     if e.type == pygame.MOUSEBUTTONDOWN and e.button == nBTN_LEFT:
406                  xd,yd = UpDate_Scroll_Mapa(nMx,nMy) # Scroll Mapa
407    Pinta_Panel()
408    UpDate_Mapa()
409    UpDate_Robots()
410    Pinta_Robots()
411    Pinta_Mapa()
412    UpDate_Naves()
413    Pinta_Naves()
414    Pinta_MMapa()
415    Pinta_Mouse()
416    pygame.display.flip()
417 
418   pygame.quit
419 
420 
421 
422 
```