

Actividad 10

CARRERA: Ingeniería en Computación

NOMBRE: *Efrain Robles Pulido*

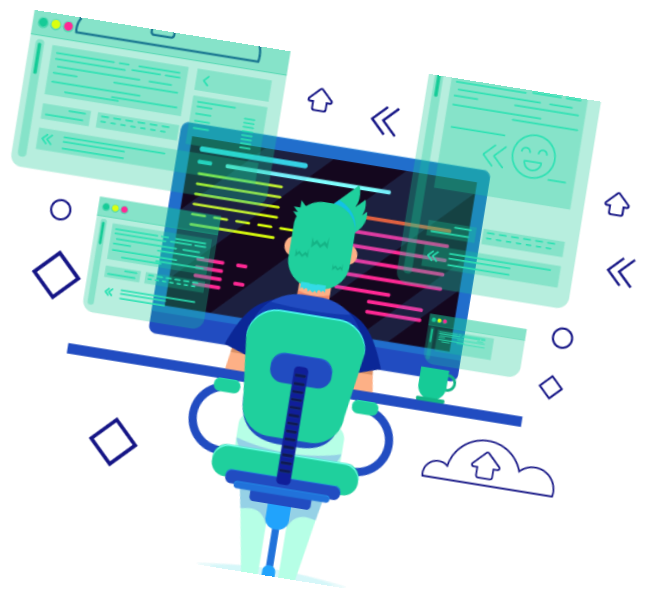
CÓDIGO: 221350095

MATERIA: Seminario de Solución de Problemas de Traductores de Lenguajes I

MAESTRA: José Juan Meza Espinoza

SECCIÓN: D09 **CALENDARIO:** 2023A

UNIVERSIDAD DE GUADALAJARA



Desarrollo

- Dibujar un rectángulo.
- Posicionar con el mouse la coordenada 1 (x0, y0) y la segunda coordenada (x1, y1).
- Dibujarlo.

El código comenzará con el almacenamiento de la dirección de memoria de nuestras variables a utilizar, como las 2 coordenadas del mouse a utilizar y definiendo una constante numérica para determinar el color del píxel a pintar, después se establecerá el modo de video en Ah en 0 y su dimensión de la pantalla en Al en 13h mediante la interrupción 10h, después moveremos en Ah el valor de 0 para ejecutar la interrupción 33h para reiniciar y verificar el driver del mouse.

A continuación, se iniciará un ciclo en el que se estará moviendo el valor de 3h en Ax con interrupción 33h para leer el estado de los botones del mouse y su respectiva posición en la pantalla de video, en el que se estará almacenando el estado de los botones en el registro Bx, y la coordenada 0's, X0 en Cx y Y0 en Dx, y a su vez se estará almacenando en memoria esas mismas coordenadas. También estaremos comparando si en Bx tenemos el valor 1 (botón izquierdo) para realizar el respectivo salto en "repite", si no es igual a 1 se estará ciclando hasta apretar el botón del mouse.

Una vez presionado el botón, pasaremos a otro ciclo en donde repetiremos el mismo procedimiento, pero ahora lo estaremos almacenando la coordenada 1's, es sus respectivos ejes, X1 en Cx y Y1 en Dx, estará saltando en "repite1" para ciclar esta parte. A continuación, movemos 0Ch a Ah para configurar para escribir un punto en pantalla y moveremos el valor COLOR en Al para configurar el color de aquel punto a pintar, después moveremos las coordenadas X0, Y0 a Cx y Dx respectivamente para empezar a pintar este punto. En el que estaremos ciclando y incrementando en 1, el valor de la coordenada X0 (Cx) para irlo pintando continuamente con la condición de que no sea igual a la coordenada X1. Pasando a otro muy parecido en el que ahora es incrementara en 1 el valor de Dx (en el eje de las Ys) y se cambiara el valor de la condición con la coordenada Y1.

El siguiente ciclo es parecido a los anteriores, pero ahora se decrementará en 1 valor de Cx (eje de la Xs) hasta llegar a la coordenada de X0, y el siguiente ciclo con decrementara en 1 el valor en Dx(eje de la Ys) hasta llegar a la coordenada Y0.

Por lo que cada ciclo tendrá su respectiva etiqueta para realizar sus ciclos individuales para formar el rectángulo. Finalmente terminara el código con la instrucción 0h de la interrupción 16h.

```

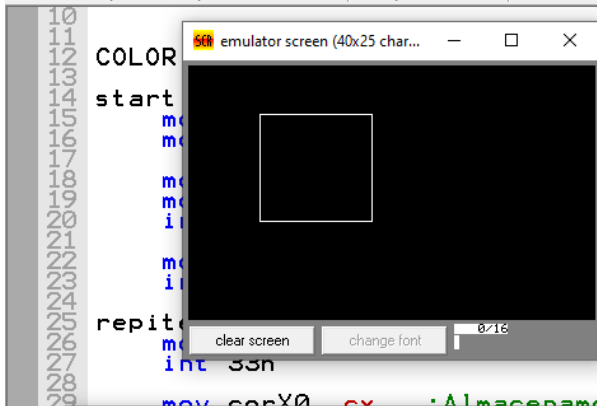
01 org 100h
02
03 jmp start
04
05 corX0 dw ?           ;Valor en memoria para la coordena X0 del mouse
06 corY0 dw ?           ;Valor en memoria para la coordena Y0 del mouse
07
08 corX1 dw ?           ;Valor en memoria para la coordena X1 del mouse
09 corY1 dw ?           ;Valor en memoria para la coordena Y1 del mouse
10
11 COLOR EQU 0Fh        ;La expresion COLOR es igual a 0Fh (255d), como constante numerica
12
13 start:
14     mov ax, @data      ;Se almacena la direccion de memoria de nuestro arreglos
15     mov ds, ax         ;Guardandola en el registro para las direcciones
16
17     mov ah, 0          ;Establece el modo video
18     mov al, 13h        ;320 x 200 en grafico
19     int 10h
20
21     mov ah, 0h         ;Verifica driver del raton
22     int 33h
23
24 repite:
25     mov ax, 03h        ;Lee el estado de los botones y la posicion del cursor
26     int 33h
27
28     mov corX0, cx       ;Almacenamos en memoria las coordenadas X0 y Y0 del mouse
29     mov corY0, dx
30
31     cmp bx, 1          ;Compara si Bx es 1 ya que significa el boton Izquierdo
32     mov bx, 0
33     jne repite         ;Si NO es igual a 1, saltara a la etiqueta repite
34
35 repite1:
36     mov ax, 03h        ;Lee el estado de los botones y la posicion del cursor
37     int 33h
38
39     mov corX1, cx       ;Almacenamos en memoria las coordenadas X1 y Y1 del mouse
40     mov corY1, dx
41
42     cmp bx, 1          ;Compara si Bx es 1 ya que significa el boton Izquierdo
43     jne repite1        ;Si NO es igual a 1, saltara a la etiqueta dibuja
44
45     mov ah, 0Ch        ;Escribira un punto en la pantalla en modo video, con lo que tenga en Cx y Dx, con la int 10h
46     mov al, COLOR      ;Valor del color a pintar
47
48     mov cx, corX0       ;Moveremos las coordenadas de X0, Y0, para empezar a dibujar la figura
49     mov dx, corY0
50
51 dibujaX0:
52     int 10h            ;Ejecucion de la interrupcion 10h, ya con sus parametros necesarios
53     inc cx             ;Incremento del registro Cx para recorrer en el eje X
54     cmp cx, corX1      ;Condicion de paro del ciclo
55     jne dibujaX0       ;Si no es igual a distancia X1, saltara a la etiqueta dibujaX0
56
57 dibujaY1:
58     int 10h            ;Ejecucion de la interrupcion 10h, ya con sus parametros necesario
59     inc dx             ;Incremento del registro Dx para recorrer en el eje Y
60     cmp dx, corY1      ;Condicion de paro del ciclo
61     jne dibujaY1       ;Si no es igual a distancia Y1, saltara a la etiqueta dibujaY1
62
63 dibujaX1:
64     int 10h            ;Ejecucion de la interrupcion 10h, ya con sus parametros necesario
65     dec cx             ;Decremento del registro Cx para recorrer en el eje X
66     cmp cx, corX0      ;Condicion de paro del ciclo
67     jne dibujaX1       ;Si no es igual a distancia X0, saltara a la etiqueta dibujaX1
68
69 dibujaY0:
70     int 10h            ;Ejecucion de la interrupcion 10h, ya con sus parametros necesario
71     dec dx             ;Decremento del registro Dx para recorrer en el eje Y
72     cmp dx, corY0      ;Condicion de paro del ciclo
73     jne dibujaY0       ;Si no es igual a distancia Y0, saltara a la etiqueta dibujaY0
74
75     mov ah, 0h         ;Termina la ejecucion del programa
76     int 16h
77
78 ret
79
80
81
82
83

```

edit: E:\OneDrive - Universidad de Guadalajara\TRADUCTORES\Seminario\ACT10\2.asm

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about



emulator: 2.com_

file math debug view external virtual devices virtual drive help

Load reload step back waiting for input stop step delay ms: 0

registers	H	L
AX	00	0F
BX	00	01
CX	00	38
DX	00	26
CS	F400	
IP	01C0	
SS	0700	
SP	FFF8	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:01C0				F400:01C0			
F41C0:	FF	255	RI	BIOS DI			
F41C1:	FF	255	RI	INT 016h			
F41C2:	CD	205	=	IRET			
F41C3:	16	022		ADD [BX + SI], A			
F41C4:	CF	207	di	ADD [BX + SI], A			
F41C5:	00	000	NI	ADD [BX + SI], A			
F41C6:	00	000	NI	ADD [BX + SI], A			
F41C7:	00	000	NI	ADD [BX + SI], A			
F41C8:	00	000	NI	ADD BH, BH			
F41C9:	00	000	NI	DEC BP			
F41CA:	00	000	NI	ADC DI, CX			
F41CB:	00	000	NI	ADD [BX + SI], A			
F41CC:	00	000	NI	ADD [BX + SI], A			
F41CD:	00	000	NI	ADD [BX + SI], A			
F41CE:	00	000	NI	ADD [BX + SI], A			
F41CF:	00	000	NI	...			

screen source reset aux vars debug stack flags

variables

size: word elements: 1

edit show as: hex

CORX0	0038h
CORY0	0026h
CORX1	0090h
CORY1	007Ah

```
cmp bx, 1          ; Compare flags
jne repite1        ; Si N

mov ah, 0Ch        ; Escribir valor
mov al, COLOR      ; Valor de color
mov cx, corX0      ; Move color X
mov dx, corY0      ; Move color Y

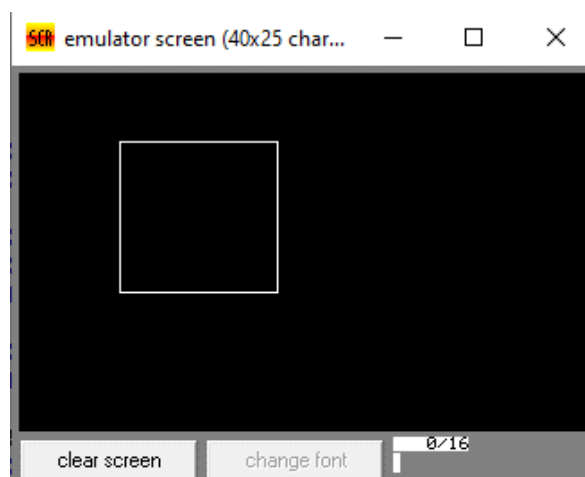
dibujaX0:
int 10h            ; Ejecucion de la interrupcion
inc cx             ; Incremento del registro CX
cmp cx, corX1      ; Condicion de paro de CX
jne dibujaX0       ; Si no es igual a dibujar X

dibujaY1:
int 10h            ; Ejecucion de la interrupcion
inc dx             ; Incremento del registro DX
cmp dx, corY1      ; Condicion de paro de DX
jne dibujaY1       ; Si no es igual a dibujar Y

dibujaX1:
int 10h            ; Ejecucion de la interrupcion
dec cx             ; Decremento del registro CX
cmp cx, corX0      ; Condicion de paro de CX
jne dibujaX1       ; Si no es igual a dibujar X
```

original source code

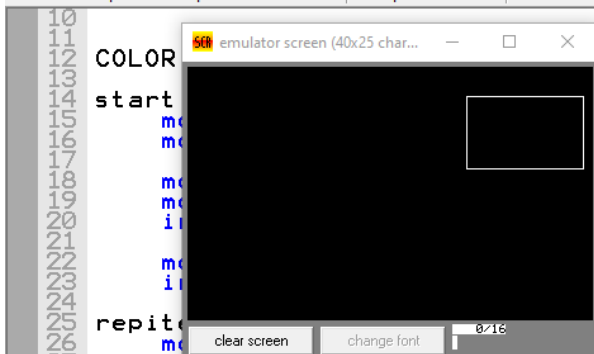
```
56 cmp cx, corX1      ; Condicion de paro de CX
57 jne dibujaX0       ; Si no es igual a dibujar X
58
59 dibujaY1:
60
61 int 10h            ; Ejecucion de la interrupcion
62 inc dx             ; Incremento del registro DX
63 cmp dx, corY1      ; Condicion de paro de DX
64 jne dibujaY1       ; Si no es igual a dibujar Y
65
66 dibujaX1:
67
68 int 10h            ; Ejecucion de la interrupcion
69 dec cx             ; Decremento del registro CX
70 cmp cx, corX0      ; Condicion de paro de CX
71 jne dibujaX1       ; Si no es igual a dibujar X
72
73 dibujaY0:
74
75 int 10h            ; Ejecucion de la interrupcion
76 dec dx             ; Decremento del registro DX
77 cmp dx, corY0      ; Condicion de paro de DX
78 jne dibujaY0       ; Si no es igual a dibujar Y
79
80 mov ah, 0h         ; Termina la ejecucion
81 int 16h
```



edit: E:\OneDrive - Universidad de Guadalajara\TRADUCTORES\Seminario\ACT10\2.asm

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about



variables

size: word elements: 1

edit show as: hex

CORX0	00DCh
CORY0	0017h
CORX1	0138h
CORY1	0050h

```
cmp bx, 1 ;Comp flags
jne repite1 ;Si N

mov ah, 0Ch ;Escri
mov al, COLOR ;Valo
mov cx, corX0 ;Move
mov dx, corY0

dibujaX0:
int 10h ;Eje
inc cx ;Inc
cmp cx, corX1 ;Con
jne dibujaX0 ;Si

dibujaY1:
int 10h ;Ejecucion d
inc dx ;Incremento
cmp dx, corY1 ;Condicio
jne dibujaY1 ;Si no es ig

dibujaX1:
```

emulator: 2.com_

file math debug view external virtual devices virtual drive help

Load reload step back waiting for input stop step delay ms: 0

registers	H	L
AX	00	0F
BX	00	01
CX	00	DC
DX	00	17
CS	F400	
IP	01C0	
SS	0700	
SP	FFF8	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

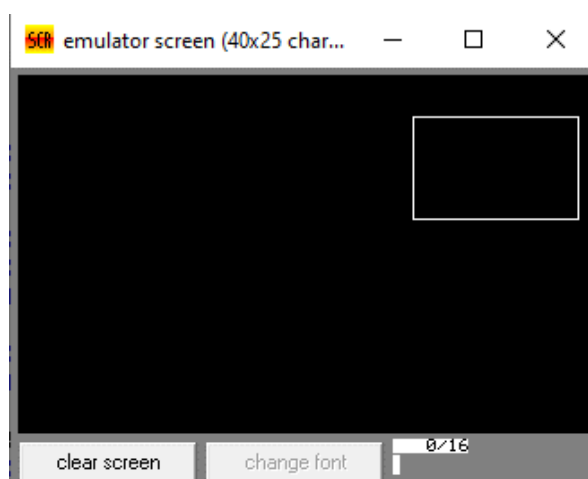
F400:01C0 F400:01C0

F41C0:	FF	255	R	BIOS DI
F41C1:	FF	255	R	INT 016h
F41C2:	CD	205	=	IRET
F41C3:	16	022		ADD [BX + SI], A
F41C4:	CF	207		ADD [BX + SI], A
F41C5:	00	000	NI	ADD [BX + SI], A
F41C6:	00	000	NI	ADD [BX + SI], A
F41C7:	00	000	NI	ADD [BX + SI], A
F41C8:	00	000	NI	ADD [BX + SI], A
F41C9:	00	000	NI	DEC BH, BH
F41CA:	00	000	NI	DEC BP
F41CB:	00	000	NI	ADC DI, CX
F41CC:	00	000	NI	ADD [BX + SI], A
F41CD:	00	000	NI	ADD [BX + SI], A
F41CE:	00	000	NI	ADD [BX + SI], A
F41CF:	00	000	NI	...

screen source reset aux vars debug stack flags

original source code

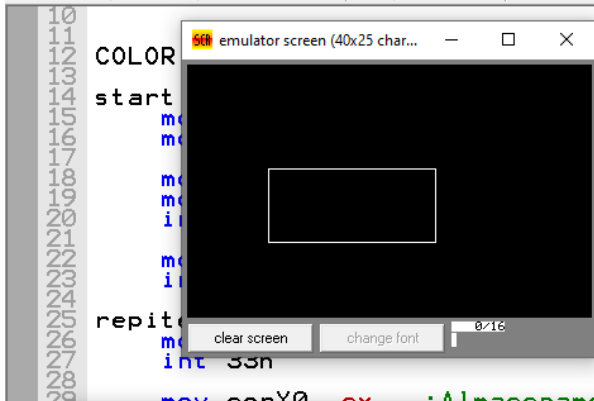
```
56 cmp cx, corX1 ;Condicion de paro de
57 jne dibujaX0 ;Si no es igual a dis
58
59 dibujaY1:
60
61 int 10h ;Ejecucion de la inte
62 inc dx ;Incremento del regis
63 cmp dx, corY1 ;Condicion de paro de
64 jne dibujaY1 ;Si no es igual a dis
65
66 dibujaX1:
67
68 int 10h ;Ejecucion de la inte
69 dec cx ;Decremento del regis
70 cmp cx, corX0 ;Condicion de paro de
71 jne dibujaX1 ;Si no es igual a dis
72
73 dibujaY0:
74
75 int 10h ;Ejecucion de la inte
76 dec dx ;Decremento del regis
77 cmp dx, corY0 ;Condicion de paro de
78 jne dibujaY0 ;Si no es igual a dis
79
80 mov ah, 0h ;Termina la ejecucion
81 int 16h
```



edit: E:\OneDrive - Universidad de Guadalajara\TRADUCTORES\Seminario\ACT10\2.asm

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about



variables

size: word elements: 1

edit show as: hex

CORX0	0040h
CORY0	0052h
CORX1	00C4h
CORY1	008Ch

```
cmp bx, 1 ;Comp flags
jne repite1 ;Si N

mov ah, 0Ch ;Escr
mov al, COLOR ;Valo

mov cx, corX0 ;Move
mov dx, corY0

dibujaX0:
int 10h ;Eje
inc cx ;Inc
cmp cx, corX1 ;Con
jne dibujaX0 ;Si

dibujaY1:
int 10h ;Ejecucion d
inc dx ;Incremento
cmp dx, corY1 ;Condicio
jne dibujaY1 ;Si no es is

dibujaX1:
```

emulator: 2.com_

file math debug view external virtual devices virtual drive help

Load reload step back waiting for input stop step delay ms: 0

registers

	H	L
AX	00	0F
BX	00	01
CX	00	40
DX	00	52
CS	F400	
IP	01C0	
SS	0700	
SP	FFF8	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

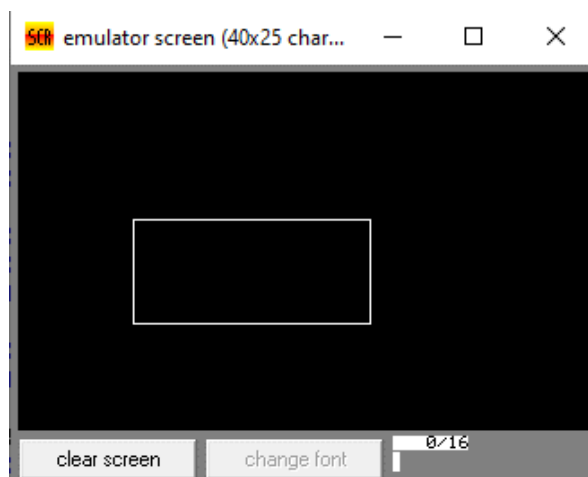
F400:01C0 F400:01C0

F41C0:	FF	255	R	BIOS DI
F41C1:	FF	255	R	INT 016h
F41C2:	CD	205	=	IRET
F41C3:	16	022		ADD [BX + SI], A
F41C4:	CF	207		ADD [BX + SI], A
F41C5:	00	000	NI	ADD [BX + SI], A
F41C6:	00	000	NI	ADD [BX + SI], A
F41C7:	00	000	NI	ADD [BX + SI], A
F41C8:	00	000	NI	ADD BH, BH
F41C9:	00	000	NI	DEC BP
F41CA:	00	000	NI	ADC DI, CX
F41CB:	00	000	NI	ADD [BX + SI], A
F41CC:	00	000	NI	ADD [BX + SI], A
F41CD:	00	000	NI	ADD [BX + SI], A
F41CE:	00	000	NI	ADD [BX + SI], A
F41CF:	00	000	NI	...

screen source reset aux vars debug stack flags

original source code

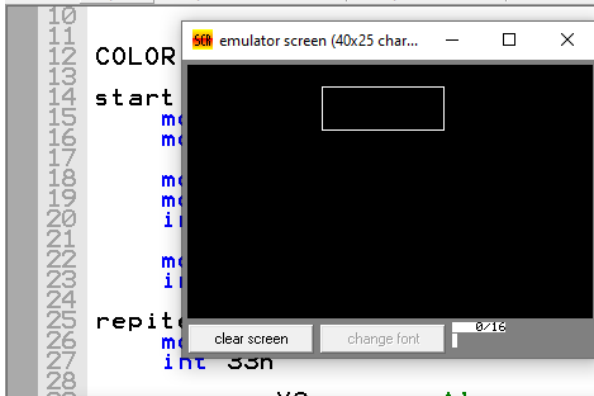
```
56 cmp cx, corX1 ;Condicion de paro de
57 jne dibujaX0 ;Si no es igual a dis
58
59 dibujaY1:
60
61 int 10h ;Ejecucion de la inte
62 inc dx ;Incremento del regis
63 cmp dx, corY1 ;Condicion de paro de
64 jne dibujaY1 ;Si no es igual a dis
65
66 dibujaX1:
67
68 int 10h ;Ejecucion de la inte
69 dec cx ;Decremento del regis
70 cmp cx, corX0 ;Condicion de paro de
71 jne dibujaX1 ;Si no es igual a dis
72
73 dibujaY0:
74
75 int 10h ;Ejecucion de la inte
76 dec dx ;Decremento del regis
77 cmp dx, corY0 ;Condicion de paro de
78 jne dibujaY0 ;Si no es igual a dis
79
80 mov ah, 0h ;Termina la ejecucion
81 int 16h
```



edit: E:\OneDrive - Universidad de Guadalajara\TRADUCTORES\Seminario\ACT10\2.asm

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about



variables

size: word elements: 1

edit show as: hex

CORX0	006Ah
CORY0	0011h
CORX1	00CAh
CORY1	0033h

```
10 COLOR
11 start
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
```

```
repite:
    mov cx, corX0
    mov dx, corY0

dibujaX0:
    int 10h
    inc cx
    cmp cx, corX1
    jne dibujaX0

dibujaY1:
    int 10h
    inc dx
    cmp dx, corY1
    jne dibujaY1

dibujaX1:
    int 10h
    dec dx
    cmp dx, corY0
    jne dibujaX1

    mov ah, 0h
    int 16h
```

emulator: 2.com_

file math debug view external virtual devices virtual drive help

Load reload step back waiting for input stop step delay ms: 0

registers

	H	L
AX	00	0F
BX	00	01
CX	00	6A
DX	00	11
CS	F400	
IP	01C0	
SS	0700	
SP	FFF8	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:01C0 F400:01C0

Address	Hex	Dec	Symbol
F41C0:	FF	255	R
F41C1:	FF	255	R
F41C2:	CD	205	=
F41C3:	16	022	
F41C4:	CF	207	
F41C5:	00	000	NI
F41C6:	00	000	NI
F41C7:	00	000	NI
F41C8:	00	000	NI
F41C9:	00	000	NI
F41CA:	00	000	NI
F41CB:	00	000	NI
F41CC:	00	000	NI
F41CD:	00	000	NI
F41CE:	00	000	NI
F41CF:	00	000	NI

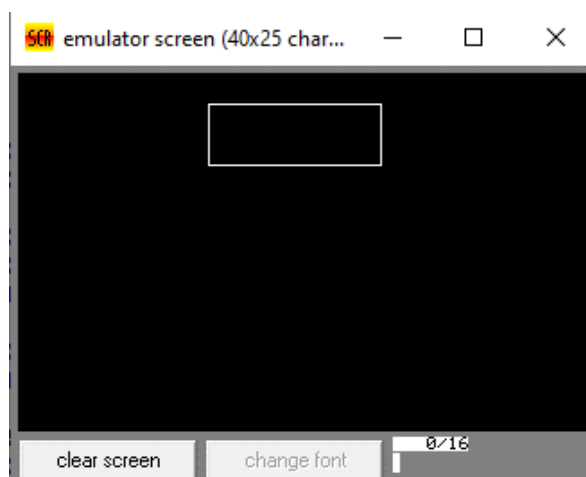
BIOS DI

```
INT 016h
IRET
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
DEC BH, BH
ADC DI, CX
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
...
```

screen source reset aux vars debug stack flags

original source code

```
56 cmp cx, corX1 ;Condicion de paro de
57 jne dibujaX0 ;Si no es igual a dis
58
59 dibujaY1:
60
61 int 10h ;Ejecucion de la inte
62 inc dx ;Incremento del regis
63 cmp dx, corY1 ;Condicion de paro de
64 jne dibujaY1 ;Si no es igual a dis
65
66 dibujaX1:
67
68 int 10h ;Ejecucion de la inte
69 dec cx ;Decremento del regis
70 cmp cx, corX0 ;Condicion de paro de
71 jne dibujaX1 ;Si no es igual a dis
72
73 dibujaY0:
74
75 int 10h ;Ejecucion de la inte
76 dec dx ;Decremento del regis
77 cmp dx, corY0 ;Condicion de paro de
78 jne dibujaY0 ;Si no es igual a dis
79
80 mov ah, 0h ;Termina la ejecucion
81 int 16h
```



Conclusión

Al principio me costo trabajo como empezar a dibujar el rectángulo. Así que primero se empezó por realizar el código para almacenar las coordenadas de ambos puntos en la memoria en el que reutilice parte del código de la práctica anterior. Para después ir aumentando en uno en uno los puntos de cada eje respectivamente. Por lo que se iba recorriendo los ejes respectivamente para dibujar el píxel correspondiente, teniendo las coordenadas almacenadas como condiciones para parar o continuar dibujando. Entonces, una vez entendido la lógica se puedo realizar correctamente la práctica, así que considero que se podría hacer aún más eficiente el código, pero con este cumple su propósito.

Bibliografía:

Brey, B. B. (2006). Microprocesadores Intel - 7 Edición (7a). Pearson Publications Company.