



ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Organización de Computadores (CCPG1049)

Implementación de Sistema de Cifrado César en Assembly x86

Integrantes:

- Luis Emmanuel Vargas Silva
- Geovanny Jahir Diaz Loor

Mgtr. Freddy Ronald Veloz De La Torre

28 de noviembre de 2025

Guayaquil - Ecuador

Contents

1 Introducción	3
2 Marco Teórico	4
2.1 Concepto del Cifrado César.....	4
2.2 Desarrollo e Implementación.....	4
2.3 Estructura de Datos y Lectura	4
2.4 Lógica de Encriptación	5
2.5 Lógica de Desencriptación y Manejo de Negativos.....	6
2.6 Caracteres Especiales.....	6
3 Manual de Usuario y Pruebas	7
3.1 Menú Principal.....	7
3.2 Proceso de Cifrado.....	7
3.3 Proceso de Descifrado.....	8
4 Conclusiones	9
5 Referencias.....	10
6 Apéndice A: Código Fuente	11
6.1 Github:	11
6.2 Código	11

1 Introducción

El presente proyecto tiene como objetivo la implementación de un sistema de cifrado y descifrado de mensajes utilizando el algoritmo César, desarrollado en lenguaje ensamblador para la arquitectura 8086. Este desarrollo responde a los requisitos de la primera evaluación de la materia Organización de Computadores, buscando reforzar el aprendizaje en la manipulación de cadenas, bucles, operaciones aritméticas y la organización modular en procedimientos dentro del entorno de bajo nivel.

El programa permite al usuario interactuar mediante una consola, ofreciendo la capacidad de procesar mensajes de hasta 35 caracteres y aplicar un desplazamiento configurable de entre 1 y 5 posiciones. Para su ejecución y validación, se utiliza el emulador emu8086, garantizando la compatibilidad con las instrucciones del microprocesador 8086.

2 Marco Teórico

2.1 Concepto del Cifrado César

El cifrado César es una técnica de criptografía simple basada en la sustitución mono-alfabética.

El algoritmo consiste en desplazar cada letra del mensaje original un número fijo de posiciones en el alfabeto. Matemáticamente, el proceso se define mediante aritmética modular para asegurar un comportamiento circular (wrap-around), donde la letra 'Z' desplazada una posición se convierte en 'A'.

La fórmula general para el cifrado es:

$$C = (P + k) \bmod 26$$

Donde:

- C es la posición de la letra cifrada (0-25).
- P es la posición de la letra original.
- k es el desplazamiento o clave .

Para el descifrado, se aplica la operación inversa: $P = (C - k) \bmod 26$

En casos donde la resta resulta en un número negativo, se suma el módulo (26) para corregir la posición.

2.2 Desarrollo e Implementación

El sistema se desarrolló utilizando una estructura modular en Assembly 8086. A continuación, se detallan los componentes críticos del código y su lógica.

2.3 Estructura de Datos y Lectura

Se definió un buffer en el segmento de datos (.DATA) capaz de almacenar 35 caracteres más un terminador \$ requerido por las interrupciones de DOS. El control de la longitud del texto se

realiza mediante un contador en el registro CX, deteniendo la lectura si se presiona la tecla ENTER (ASCII 13) o si se alcanza el límite de caracteres.

Fragmento de código

; Fragmento de validación de longitud (*Ver Apéndice A*)

cmp cx, 35 ; límite de 35 caracteres

jl LEER_TEXTO ; Si es menor, continua leyendo

2.4 Lógica de Encriptación

El núcleo del procesamiento distingue entre letras mayúsculas, minúsculas y caracteres no alfabéticos para preservar la integridad del mensaje.

1. **Normalización:** Debido a que los códigos ASCII de las letras no comienzan en 0 (A=65, a=97), es necesario normalizar el valor restando la base ASCII antes de operar.
2. **Operación Modular:** Se suma el desplazamiento y se calcula el residuo de la división por 26 para manejar la circularidad del alfabeto.
3. **Desnormalización:** Se suma la base ASCII original al resultado para obtener el carácter cifrado final.

El siguiente segmento de código ilustra el tratamiento para mayúsculas:

Fragmento de código

ES_MAYUSCULA:

sub al, 'A' ; normalizar a 0-25

add al, bl ; sumar desplazamiento

mov ah, 0

mov cl, 26

```
div cl      ; dividir entre 26  
mov al, ah  ; tomar residuo (modulo 26)  
add al, 'A'  ; volver a ASCII
```

(Véase el código completo en Apéndice A)

2.5 Lógica de Desencriptación y Manejo de Negativos

En el proceso inverso, al restar el desplazamiento, el resultado puede ser negativo (ej. 'A' menos 3 posiciones). La implementación verifica el signo del resultado; si es negativo, se suma 26 para rotar al final del alfabeto.

Fragmento de código

ES_MAYUSCULA DES:

```
sub al, 'A'  
sub al, bl  ; restar desplazamiento  
cmp al, 0   ; verificar si es negativo  
jge POSITIVO_MAY  
add al, 26  ; ajustar si es negativo (vuelta atrás)
```

2.6 Caracteres Especiales

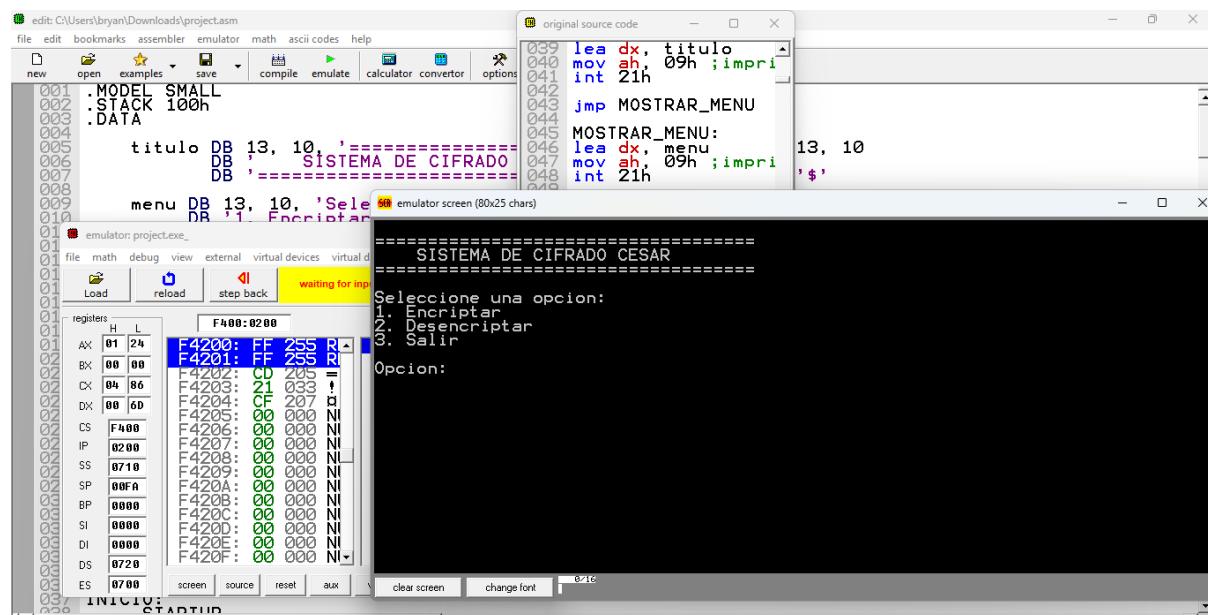
El sistema implementa una política de preservación para cualquier carácter que no sea una letra (espacios, números, signos). Estos se imprimen directamente sin modificaciones, saltando la lógica de cifrado.

3 Manual de Usuario y Pruebas

3.1 Menú Principal

Al iniciar el programa en emu8086, se presenta un menú interactivo con tres opciones. El sistema valida que el usuario ingrese únicamente los números '1', '2' o '3'.

Captura 1: Menú Principal



3.2 Proceso de Cifrado

El usuario selecciona la opción '1', ingresa un desplazamiento (validado entre 1 y 5) y posteriormente el texto.

Caso de Prueba:

- **Entrada:** HOLA MUNDO
- **Desplazamiento:** 3
- **Resultado Esperado:** KROD PXQGR.

Captura 2: Ejecución de Cifrado

The screenshot shows a debugger interface with two windows. The left window displays assembly code for a program named 'project.asm'. The right window shows the terminal output of the program's execution.

Assembly Code (Left Window):

```
001 .MODEL SMALL
002 .STACK 100h
003 .DATA
004     titulo DB 13, 10, '===== SISTEMA DE CIFRADO ====='
005     DB ' '
006     DB ' '
007     menu DB 13, 10, 'Selección: '
008     DB '1. Encriptar'
009     DB '2. Desencriptar'
010     DB '3. Salir'
011
012 emulator:project.exe_
013 file matte debug view external virtual devices virtual d
014 Load [ ] reload [ ] step back [ ] waiting for input [ ]
015
016 registers H L
017 AX 01 24
018 BX 00 03
019 CX 00 1A
020 DX 01 35
021 CS F400
022 IP 0200
023 SS 0710
024 SP 00FA
025 BP 0000
026 DI 0000
027 SI 01C8
028 DS 0720
029 ES 0700
030
031 INICIO: ORG 100H
032
033 mov ah, 02h ;imprime el caracter
034 int 21h
035 SIGUIENTE_CARACTER
036 inc si ;siguiente
037 jmp PROCESAR_CARAC
038 FIN_ENCRYPTADO: 13, 10
039 lea dx, presione_t
040 mov ah, 09h
041 int 21h
042
043
044
045
046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
0100
0101
0102
0103
0104
0105
0106
0107
0108
0109
0110
0111
0112
0113
0114
0115
0116
0117
0118
0119
0120
0121
0122
0123
0124
0125
0126
0127
0128
0129
0130
0131
0132
0133
0134
0135
0136
0137
0138
0139
0140
0141
0142
0143
0144
0145
0146
0147
0148
0149
0150
0151
0152
0153
0154
0155
0156
0157
0158
0159
0160
0161
0162
0163
0164
0165
0166
0167
0168
0169
0170
0171
0172
0173
0174
0175
0176
0177
0178
0179
0180
0181
0182
0183
0184
0185
0186
0187
0188
0189
0190
0191
0192
0193
0194
0195
0196
0197
0198
0199
0200
0201
0202
0203
0204
0205
0206
0207
0208
0209
0210
0211
0212
0213
0214
0215
0216
0217
0218
0219
0220
0221
0222
0223
0224
0225
0226
0227
0228
0229
0230
0231
0232
0233
0234
0235
0236
0237
0238
0239
0240
0241
0242
0243
0244
0245
0246
0247
0248
0249
0250
0251
0252
0253
0254
0255
0256
0257
0258
0259
0260
0261
0262
0263
0264
0265
0266
0267
0268
0269
0270
0271
0272
0273
0274
0275
0276
0277
0278
0279
0280
0281
0282
0283
0284
0285
0286
0287
0288
0289
0290
0291
0292
0293
0294
0295
0296
0297
0298
0299
0300
0301
0302
0303
0304
0305
0306
0307
0308
0309
0310
0311
0312
0313
0314
0315
0316
0317
0318
0319
0320
0321
0322
0323
0324
0325
0326
0327
0328
0329
0330
0331
0332
0333
0334
0335
0336
0337
0338
0339
0340
0341
0342
0343
0344
0345
0346
0347
0348
0349
0350
0351
0352
0353
0354
0355
0356
0357
0358
0359
0360
0361
0362
0363
0364
0365
0366
0367
0368
0369
0370
0371
0372
0373
0374
0375
0376
0377
0378
0379
0380
0381
0382
0383
0384
0385
0386
0387
0388
0389
0390
0391
0392
0393
0394
0395
0396
0397
0398
0399
0400
0401
0402
0403
0404
0405
0406
0407
0408
0409
0410
0411
0412
0413
0414
0415
0416
0417
0418
0419
0420
0421
0422
0423
0424
0425
0426
0427
0428
0429
0430
0431
0432
0433
0434
0435
0436
0437
0438
0439
0440
0441
0442
0443
0444
0445
0446
0447
0448
0449
0450
0451
0452
0453
0454
0455
0456
0457
0458
0459
0460
0461
0462
0463
0464
0465
0466
0467
0468
0469
0470
0471
0472
0473
0474
0475
0476
0477
0478
0479
0480
0481
0482
0483
0484
0485
0486
0487
0488
0489
0490
0491
0492
0493
0494
0495
0496
0497
0498
0499
0500
0501
0502
0503
0504
0505
0506
0507
0508
0509
0510
0511
0512
0513
0514
0515
0516
0517
0518
0519
0520
0521
0522
0523
0524
0525
0526
0527
0528
0529
0530
0531
0532
0533
0534
0535
0536
0537
0538
0539
0540
0541
0542
0543
0544
0545
0546
0547
0548
0549
0550
0551
0552
0553
0554
0555
0556
0557
0558
0559
0560
0561
0562
0563
0564
0565
0566
0567
0568
0569
0570
0571
0572
0573
0574
0575
0576
0577
0578
0579
0580
0581
0582
0583
0584
0585
0586
0587
0588
0589
0589
0590
0591
0592
0593
0594
0595
0596
0597
0598
0599
0600
0601
0602
0603
0604
0605
0606
0607
0608
0609
0610
0611
0612
0613
0614
0615
0616
0617
0618
0619
0620
0621
0622
0623
0624
0625
0626
0627
0628
0629
0630
0631
0632
0633
0634
0635
0636
0637
0638
0639
0640
0641
0642
0643
0644
0645
0646
0647
0648
0649
0650
0651
0652
0653
0654
0655
0656
0657
0658
0659
0660
0661
0662
0663
0664
0665
0666
0667
0668
0669
0669
0670
0671
0672
0673
0674
0675
0676
0677
0678
0679
0679
0680
0681
0682
0683
0684
0685
0686
0687
0688
0689
0689
0690
0691
0692
0693
0694
0695
0696
0697
0698
0699
0700
0701
0702
0703
0704
0705
0706
0707
0708
0709
0709
0710
0711
0712
0713
0714
0715
0716
0717
0718
0719
0719
0720
0721
0722
0723
0724
0725
0726
0727
0728
0729
0729
0730
0731
0732
0733
0734
0735
0736
0737
0738
0739
0739
0740
0741
0742
0743
0744
0745
0746
0747
0748
0749
0749
0750
0751
0752
0753
0754
0755
0756
0757
0758
0759
0759
0760
0761
0762
0763
0764
0765
0766
0767
0768
0769
0769
0770
0771
0772
0773
0774
0775
0776
0777
0778
0779
0779
0780
0781
0782
0783
0784
0785
0786
0787
0788
0789
0789
0790
0791
0792
0793
0794
0795
0796
0797
0798
0799
0800
0801
0802
0803
0804
0805
0806
0807
0808
0809
0809
0810
0811
0812
0813
0814
0815
0816
0817
0818
0819
0819
0820
0821
0822
0823
0824
0825
0826
0827
0828
0829
0829
0830
0831
0832
0833
0834
0835
0836
0837
0838
0839
0839
0840
0841
0842
0843
0844
0845
0846
0847
0848
0849
0849
0850
0851
0852
0853
0854
0855
0856
0857
0858
0859
0859
0860
0861
0862
0863
0864
0865
0866
0867
0868
0869
0869
0870
0871
0872
0873
0874
0875
0876
0877
0878
0879
0879
0880
0881
0882
0883
0884
0885
0886
0887
0888
0889
0889
0890
0891
0892
0893
0894
0895
0896
0897
0898
0899
0900
0901
0902
0903
0904
0905
0906
0907
0908
0909
0909
0910
0911
0912
0913
0914
0915
0916
0917
0918
0919
0919
0920
0921
0922
0923
0924
0925
0926
0927
0928
0929
0929
0930
0931
0932
0933
0934
0935
0936
0937
0938
0939
0939
0940
0941
0942
0943
0944
0945
0946
0947
0948
0949
0949
0950
0951
0952
0953
0954
0955
0956
0957
0958
0959
0959
0960
0961
0962
0963
0964
0965
0966
0967
0968
0969
0969
0970
0971
0972
0973
0974
0975
0976
0977
0978
0979
0979
0980
0981
0982
0983
0984
0985
0986
0987
0988
0989
0989
0990
0991
0992
0993
0994
0995
0996
0997
0998
0999
0100
0101
0102
0103
0104
0105
0106
0107
0108
0109
0110
0111
0112
0113
0114
0115
0116
0117
0118
0119
0120
0121
0122
0123
0124
0125
0126
0127
0128
0129
0130
0131
0132
0133
0134
0135
0136
0137
0138
0139
0140
0141
0142
0143
0144
0145
0146
0147
0148
0149
0150
0151
0152
0153
0154
0155
0156
0157
0158
0159
0160
0161
0162
0163
0164
0165
0166
0167
0168
0169
0170
0171
0172
0173
0174
0175
0176
0177
0178
0179
0180
0181
0182
0183
0184
0185
0186
0187
0188
0189
0190
0191
0192
0193
0194
0195
0196
0197
0198
0199
0200
0201
0202
0203
0204
0205
0206
0207
0208
0209
0210
0211
0212
0213
0214
0215
0216
0217
0218
0219
0220
0221
0222
0223
0224
0225
0226
0227
0228
0229
0229
0230
0231
0232
0233
0234
0235
0236
0237
0238
0239
0239
0240
0241
0242
0243
0244
0245
0246
0247
0248
0249
0249
0250
0251
0252
0253
0254
0255
0256
0257
0258
0259
0259
0260
0261
0262
0263
0264
0265
0266
0267
0268
0269
0269
0270
0271
0272
0273
0274
0275
0276
0277
0278
0279
0279
0280
0281
0282
0283
0284
0285
0286
0287
0288
0289
0289
0290
0291
0292
0293
0294
0295
0296
0297
0298
0299
0299
0300
0301
0302
0303
0304
0305
0306
0307
0308
0309
0309
0310
0311
0312
0313
0314
0315
0316
0317
0318
0319
0319
0320
0321
0322
0323
0324
0325
0326
0327
0328
0329
0329
0330
0331
0332
0333
0334
0335
0336
0337
0338
0339
0339
0340
0341
0342
0343
0344
0345
0346
0347
0348
0349
0349
0350
0351
0352
0353
0354
0355
0356
0357
0358
0359
0359
0360
0361
0362
0363
0364
0365
0366
0367
0368
0369
0369
0370
0371
0372
0373
0374
0375
0376
0377
0378
0379
0379
0380
0381
0382
0383
0384
0385
0386
0387
0388
0389
0389
0390
0391
0392
0393
0394
0395
0396
0397
0398
0399
0399
0400
0401
0402
0403
0404
0405
0406
0407
0408
0409
0409
0410
0411
0412
0413
0414
0415
0416
0417
0418
0419
0419
0420
0421
0422
0423
0424
0425
0426
0427
0428
0429
0429
0430
0431
0432
0433
0434
0435
0436
0437
0438
0439
0439
0440
0441
0442
0443
0444
0445
0446
0447
0448
0449
0449
0450
0451
0452
0453
0454
0455
0456
0457
0458
0459
0459
0460
0461
0462
0463
0464
0465
0466
0467
0468
0469
0469
0470
0471
0472
0473
0474
0475
0476
0477
0478
0479
0479
0480
0481
0482
0483
0484
0485
0486
0487
0488
0489
0489
0490
0491
0492
0493
0494
0495
0496
0497
0498
0499
0499
0500
0501
0502
0503
0504
0505
0506
0507
0508
0509
0509
0510
0511
0512
0513
0514
0515
0516
0517
0518
0519
0519
0520
0521
0522
0523
0524
0525
0526
0527
0528
0529
0529
0530
0531
0532
0533
0534
0535
0536
0537
0538
0539
0539
0540
0541
0542
0543
0544
0545
0546
0547
0548
0549
0549
0550
0551
0552
0553
0554
0555
0556
0557
0558
0559
0559
0560
0561
0562
0563
0564
0565
0566
0567
0568
0569
0569
0570
0571
0572
0573
0574
0575
0576
0577
0578
0579
0579
0580
0581
0582
0583
0584
0585
0586
0587
0588
0589
0589
0590
0591
0592
0593
0594
0595
0596
0597
0598
0599
0599
0600
0601
0602
0603
0604
0605
0606
0607
0608
0609
0609
0610
0611
0612
0613
0614
0615
0616
0617
0618
0619
0619
0620
0621
0622
0623
0624
0625
0626
0627
0628
0629
0629
0630
0631
0632
0633
0634
0635
0636
0637
0638
0639
0639
0640
0641
0642
0643
0644
0645
0646
0647
0648
0649
0649
0650
0651
0652
0653
0654
0655
0656
0657
0658
0659
0659
0660
0661
0662
0663
0664
0665
0666
0667
0668
0669
0669
0670
0671
0672
0673
0674
0675
0676
0677
0678
0679
0679
0680
0681
0682
0683
0684
0685
0686
0687
0688
0689
0689
0690
0691
0692
0693
0694
0695
0696
0697
0698
0699
0699
0700
0701
0702
0703
0704
0705
0706
0707
0708
0709
0709
0710
0711
0712
0713
0714
0715
0716
0717
0718
0719
0719
0720
0721
0722
0723
0724
0725
0726
0727
0728
0729
0729
0730
0731
0732
0733
0734
0735
0736
0737
0738
0739
0739
0740
0741
0742
0743
0744
0745
0746
0747
0748
0749
0749
0750
0751
0752
0753
0754
0755
0756
0757
0758
0759
0759
0760
0761
0762
0763
0764
0765
0766
0767
0768
0769
0769
0770
0771
0772
0773
0774
0775
0776
0777
0778
0779
0779
0780
0781
0782
0783
0784
0785
0786
0787
0788
0789
0789
0790
0791
0792
0793
0794
0795
0796
0797
0798
0799
0799
0800
0801
0802
0803
0804
0805
0806
0807
0808
0809
0809
0810
0811
0812
0813
0814
0815
0816
0817
0818
0819
0819
0820
0821
0822
0823
0824
0825
0826
0827
0828
0829
0829
0830
0831
0832
0833
0834
0835
0836
0837
0838
0839
0839
0840
0841
0842
0843
0844
0845
0846
0847
0848
0849
0849
0850
0851
0852
0853
0854
0855
0856
0857
0858
0859
0859
0860
0861
0862
0863
0864
0865
0866
0867
0868
0869
0869
0870
0871
0872
0873
0874
0875
0876
0877
0878
0879
0879
0880
0881
0882
0883
0884
0885
0886
0887
0888
0889
0889
0890
0891
0892
0893
0894
0895
0896
0897
0898
0899
0899
0900
0901
0902
0903
0904
0905
0906
0907
0908
0909
0909
0910
0911
0912
0913
0914
0915
0916
0917
0918
0919
0919
0920
0921
0922
0923
0924
0925
0926
0927
0928
0929
0929
0930
0931
0932
0933
0934
0935
0936
0937
0938
0939
0939
0940
0941
0942
0943
0944
0945
0946
0947
0948
0949
0949
0950
0951
0952
0953
0954
0955
0956
0957
0958
0959
0959
0960
0961
0962
0963
0964
0965
0966
0967
0968
0969
0969
0970
0971
0972
0973
0974
0975
0976
0977
0978
0979
0979
0980
0981
0982
0983
0984
0985
0986
0987
0988
0989
0989
0990
0991
0992
0993
0994
0995
0996
0997
0998
0999
0100
0101
0102
0103
0104
0105
0106
0107
0108
0109
0110
0111
0112
0113
0114
0115
0116
0117
0118
0119
0119
0120
0121
0122
0123
0124
0125
0126
0127
0128
0129
0129
0130
0131
0132
0133
0134
0135
0136
0137
0138
0139
0139
0140
0141
0142
0143
0144
0145
0146
0147
0148
0149
0149
0150
0151
0152
0153
0154
0155
0156
0157
0158
0159
0159
0160
0161
0162
0163
0164
0165
0166
0167
0168
0169
0169
0170
0171
0172
0173
0174
0175
0176
0177
0178
0179
0179
0180
0181
0182
0183
0184
0185
0186
0187
0188
0189
0189
0190
0191
0192
0193
0194
0195
0196
0197
0198
0199
0200
0201
0202
0203
0204
0205
0206
0207
0208
0209
0209
0210
0211
0212
0213
0214
0215
0216
0217
0218
0219
0219
0220
0221
0222
0223
0224
0225
0226
0227
0228
0229
0229
0230
0231
0232
0233
0234
0235
0236
0237
0238
0239
0239
0240
0241
0242
0243
0244
0245
0246
0247
0248
0249
0249
0250
0251
0252
02
```

4 Conclusiones

La implementación del Cifrado César en Assembly x86 permitió comprender a profundidad el manejo de registros y la aritmética a nivel de byte. Se logró cumplir con todos los requisitos funcionales, incluyendo la validación de límites de buffer (35 caracteres) y el manejo correcto del desbordamiento del alfabeto (wrap-around) mediante la división entera DIV para obtener el módulo.

Si bien el algoritmo presenta limitaciones inherentes de seguridad y un rango de desplazamiento restringido por diseño (1-5), el proyecto demuestra eficazmente la capacidad de manipular datos ASCII y controlar el flujo del programa mediante saltos condicionales y comparaciones.

5 Referencias

1. Escuela Superior Politécnica del Litoral. (2025). *Documentación del Proyecto: Organización de Computadores 2025-2*. Guayaquil: ESPOL.
2. Caesar Cipher Visualizer. (s.f.). Recuperado de <https://caesar-cipher.com/>
3. emu8086 Documentation. (s.f.). Recuperado de <http://www.emu8086.com/>

6 Apéndice A: Código Fuente

6.1 Github:

<https://github.com/g30dl/emu8086-caesar-encryption.git>

6.2 Código

A continuación, se presenta el código completo desarrollado para el proyecto:

Fragmento de código

.MODEL SMALL

.STACK 100h

.DATA

titulo DB 13, 10, '=====', 13, 10

DB ' SISTEMA DE CIFRADO CESAR', 13, 10

DB =====, 13, 10, '\$'

menu DB 13, 10, 'Seleccione una opcion:', 13, 10

DB '1. Encriptar', 13, 10

DB '2. Desencriptar', 13, 10

DB '3. Salir', 13, 10

DB 13, 10, 'Opcion: \$'

error_opcion DB 13, 10, 'Error: Opcion invalida!', 13, 10

DB 'Por favor ingrese un numero entre 1 y 3.', 13, 10, '\$'

msg_salida DB 13, 10, 13, 10, 'Gracias por usar el sistema!', 13, 10

DB 'Hasta pronto...', 13, 10, '\$'

presione_tecla DB 13, 10, 'Presione cualquier tecla para continuar...\$',

msg_desplazamiento DB 13, 10, 'Ingrese desplazamiento (1-5): \$'

msg_texto DB 'Ingrese texto (max 35 caracteres): \$'

```
msg_resultado DB 13, 10, 13, 10, 'Texto encriptado: $'
```

```
texto_entrada DB 36 DUP('$') ;buffer para 35 caracteres + terminador
```

```
desplazamiento DB 0
```

```
desplazamiento_inv DB 0
```

```
.CODE
```

```
INICIO:
```

```
.STARTUP
```

```
lea dx, titulo
```

```
mov ah, 09h ;imprimir titulo
```

```
int 21h
```

```
jmp MOSTRAR_MENU
```

```
MOSTRAR_MENU:
```

```
lea dx, menu
```

```
mov ah, 09h ;imprimir el menu
```

```
int 21h
```

```
mov ah, 01h ;leer caracter
```

```
int 21h
```

```
cmp al, '1' ;validar rango 1-3
```

```
jl OPCION_INVALIDA
```

```
cmp al, '3'
```

```
jg OPCION_INVALIDA
```

```
cmp al, '1'  
je ENcriptar  
cmp al, '2'  
je Desencriptar  
cmp al, '3'  
je Salir  
  
jmp OPCION_INVALIDA
```

OPCION_INVALIDA:

```
lea dx, error_opcion  
mov ah, 09h ;imprimir cadena  
int 21h
```

```
lea dx, presione_tecla  
mov ah, 09h  
int 21h
```

```
mov ah, 01h ;esperar tecla  
int 21h
```

```
jmp MOSTRAR_MENU
```

```
;===== ENcriptacion
```

ENcriptar:

```
lea dx, msg_desplazamiento  
mov ah, 09h ;imprimir cadena  
int 21h
```

```
mov ah, 01h ;leer caracter
```

```
int 21h  
sub al, '0' ;convertir ASCII a numero  
mov desplazamiento, al
```

```
cmp al, 1 ;validar rango 1-5  
jl MOSTRAR_MENU  
cmp al, 5  
jg MOSTRAR_MENU
```

```
lea dx, msg_texto  
mov ah, 09h ;imprimir cadena  
int 21h
```

```
lea si, texto_entrada ;SI apunta al buffer  
mov cx, 0 ;contador de caracteres
```

```
LEER_TEXTO:  
mov ah, 01h ;leer caracter  
int 21h
```

```
cmp al, 13 ;verificar ENTER  
je FIN_LECTURA
```

```
mov [si], al ;guardar en buffer  
inc si  
inc cx  
cmp cx, 35 ;limite de 35 caracteres  
jl LEER_TEXTO
```

```
FIN_LECTURA:  
mov byte ptr [si], '$' ;terminar cadena
```

```
lea dx, msg_resultado  
mov ah, 09h ;imprimir cadena  
int 21h
```

```
lea si, texto_entrada  
mov bl, desplazamiento
```

PROCESAR_CARACTER:

```
mov al, [si] ;cargar caracter actual  
cmp al, '$' ;fin de cadena  
je FIN_ENcriptado
```

```
cmp al, 'A' ;verificar si es mayuscula A-Z  
jl NO_ES_LETRA  
cmp al, 'Z'  
jle ES_MAYUSCULA
```

```
cmp al, 'a' ;verificar si es minuscula a-z  
jl NO_ES_LETRA  
cmp al, 'z'  
jle ES_MINUSCULA
```

NO_ES_LETRA:

```
mov dl, al  
mov ah, 02h ;imprimir caracter sin cambios  
int 21h  
jmp SIGUIENTE_CARACTER
```

ES_MAYUSCULA:

```
sub al, 'A' ;normalizar a 0-25  
add al, bl ;sumar desplazamiento  
mov ah, 0
```

```
mov cl, 26
div cl ;dividir entre 26
mov al, ah ;tomar residuo (modulo 26)
add al, 'A' ;volver a ASCII
mov dl, al
mov ah, 02h ;imprimir caracter
int 21h
jmp SIGUIENTE_CARACTER
```

ES_MINUSCULA:

```
sub al, 'a' ;normalizar a 0-25
add al, bl ;sumar desplazamiento
mov ah, 0
mov cl, 26
div cl ;dividir entre 26
mov al, ah ;tomar residuo (modulo 26)
add al, 'a' ;volver a ASCII
mov dl, al
mov ah, 02h ;imprimir caracter
int 21h
```

SIGUIENTE_CARACTER:

```
inc si ;siguiente caracter
jmp PROCESAR_CARACTER
```

FIN_ENcriptado:

```
lea dx, presione_tecla
mov ah, 09h
int 21h
mov ah, 01h ;esperar tecla
int 21h
jmp MOSTRAR_MENU
```

```
=====
=====
```

```
===== DESENCRIPTAR
=====
```

DESENCRIPTAR:

```
lea dx, msg_desplazamiento  
mov ah, 09h ;imprimir cadena  
int 21h
```

```
mov ah, 01h ;leer caracter  
int 21h  
sub al, '0' ;convertir ASCII a numero  
mov desplazamiento, al
```

```
cmp al, 1 ;validar rango 1-5  
jl MOSTRAR_MENU  
cmp al, 5  
jg MOSTRAR_MENU
```

```
lea dx, msg_texto  
mov ah, 09h ;imprimir cadena  
int 21h
```

```
lea si, texto_entrada ;SI apunta al buffer  
mov cx, 0 ;contador de caracteres
```

LEER_TEXTO DES:

```
mov ah, 01h ;leer caracter  
int 21h  
cmp al, 13 ;verificar ENTER
```

```
je FIN_LECTURA_DES  
mov [si], al ;guardar en buffer  
inc si  
inc cx  
cmp cx, 35 ;limite de 35 caracteres  
jl LEER_TEXTO_DES
```

FIN_LECTURA_DES:

```
mov byte ptr [si], '$' ;terminar cadena
```

```
lea dx, msg_resultado  
mov ah, 09h ;imprimir cadena  
int 21h
```

```
lea si, texto_entrada  
mov bl, desplazamiento
```

PROCESAR_DES:

```
mov al, [si] ;cargar caracter actual  
cmp al, '$' ;fin de cadena  
je FIN_DESENCRIPTADO
```

```
cmp al, 'A' ;verificar si es mayuscula A-Z  
jl NO_ES_LETRA_DES  
cmp al, 'Z'  
jle ES_MAYUSCULA_DES
```

```
cmp al, 'a' ;verificar si es minuscula a-z  
jl NO_ES_LETRA_DES  
cmp al, 'z'  
jle ES_MINUSCULA_DES
```

NO_ES_LETRA_DES:

```
mov dl, al  
mov ah, 02h ;imprimir caracter sin cambios  
int 21h  
jmp SIGUIENTE_DES
```

ES_MAYUSCULA_DES:

```
sub al, 'A' ;normalizar a 0-25  
sub al, bl ;restar desplazamiento  
  
cmp al, 0 ;verificar si es negativo  
jge POSITIVO_MAY  
add al, 26 ;ajustar si es negativo
```

POSITIVO_MAY:

```
add al, 'A' ;volver a ASCII  
mov dl, al  
mov ah, 02h ;imprimir caracter  
int 21h  
jmp SIGUIENTE_DES
```

ES_MINUSCULA_DES:

```
sub al, 'a' ;normalizar a 0-25  
sub al, bl ;restar desplazamiento  
  
cmp al, 0 ;verificar si es negativo  
jge POSITIVO_MIN  
add al, 26 ;ajustar si es negativo
```

POSITIVO_MIN:

```
add al, 'a' ;volver a ASCII  
mov dl, al
```

```
mov ah, 02h ;imprimir caracter  
int 21h
```

SIGUIENTE_DES:

```
inc si ;siguiente caracter  
jmp PROCESAR_DES
```

FIN_DESENCRIPTADO:

```
lea dx, presione_tecla  
mov ah, 09h  
int 21h  
mov ah, 01h ;esperar tecla  
int 21h  
jmp MOSTRAR_MENU
```

;

SALIR:

```
lea dx, msg_salida  
mov ah, 09h ;imprimir cadena  
int 21h  
  
mov ah, 4Ch ;terminar programa  
int 21h
```

END INICIO