

Öteleme Komutları

JHL ve JAL (ikisi de aynı)



Mov AH 4

JHL AH, 1 → 8 → Sayıyi sola

ötelemek sayıyı

2 ile çarpmak

demek olur sağa

doğru ötelemek 2'ye

bölmek demektir

JHL AH, X → Buraya thorciande
bir sayı yazamıyorumsunuz

Mov CL, 2
JHL AH, CL } Böyle yapabilir-
siniz

JHR



Mov AX 15

JHR AX. ① → Burdada aynı durum geçerli
buraya t horizonde birsey
yazamıyoruz

→ Sonuç 7

Küsürlü bölme
yapamıyoruz kalan
umrumuzda değil

SORU

Bir tane sayı depistenimiz

yar

" " Sayı "

Sayımız çift ise
AX değerini 0
yapın tek ise
t yapın assembly
Kodunu yazın

① → eger en onlamsız
biti t ise tektir
0 ise çifttir

JHR soyi, 1

JCF tek

Mov AX, 1

JMP SONUC

Mov AX, 0

JHR DADDR, idata¹

Mov AX, soyi } eger yukardaki
JHR AX, 1 } gibi bir tanim
olmasaydi böyle
yapmol 20unday-
dit

JAR



Mov AH, 10101100 B

JAR AH, 1

JAR 10101100 B

11010110

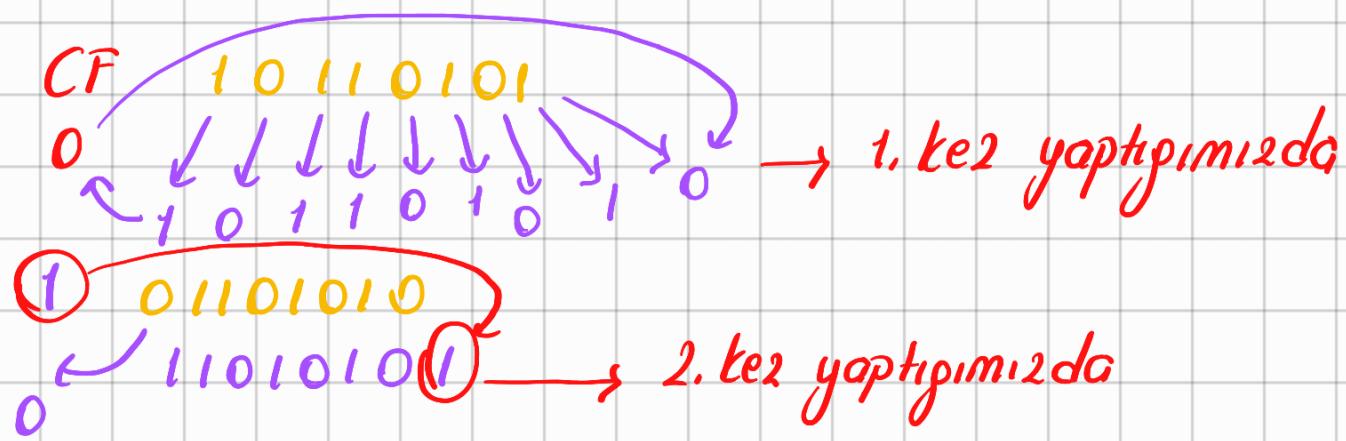
igorelli soyilorda JAR kullanırız

RCL



AH \leftarrow 10110101 CF \leftarrow 0

RCL AH,1



JTC → CF ← 1

CLC → CF ← 0

RCR



Birden fazla
rotate yapmak
muhakkak CL
registeri üzerinde
yapmamız lazım

JTC j CF=1

mov BL, 11100111B

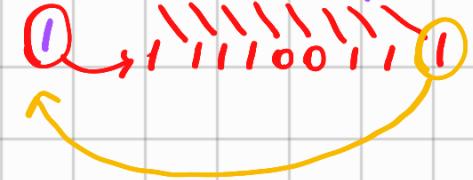
mov CL, 3

RCR BL, CL

JTC CF, 1

Boyle bir şey
yok

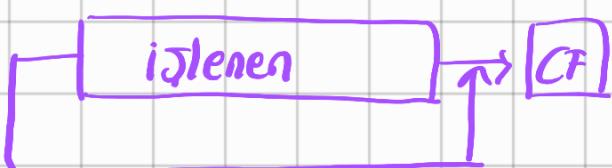
CF 11100111



ROL



ROR



$A = 111\ 00000 \rightarrow 224$ } 224'ü
 $00000\ 111 \rightarrow ?$ } ? olarak
 elde etmeye
 geliyoruz

Mov AL, 0
Mov CX, 8 → Burdaki
 LI: JHR AL, 1 döngünün
 RCL AH, 1 kaç kere
 Loop LI dönüpçepini
 Mov O, AH CX belirliyor

?

Dizgi (string), yığın (stack) komutları

MovSB

CmpSB

JcASB

LoDSB

StoSB

CBw

CWD

POP

PUSH

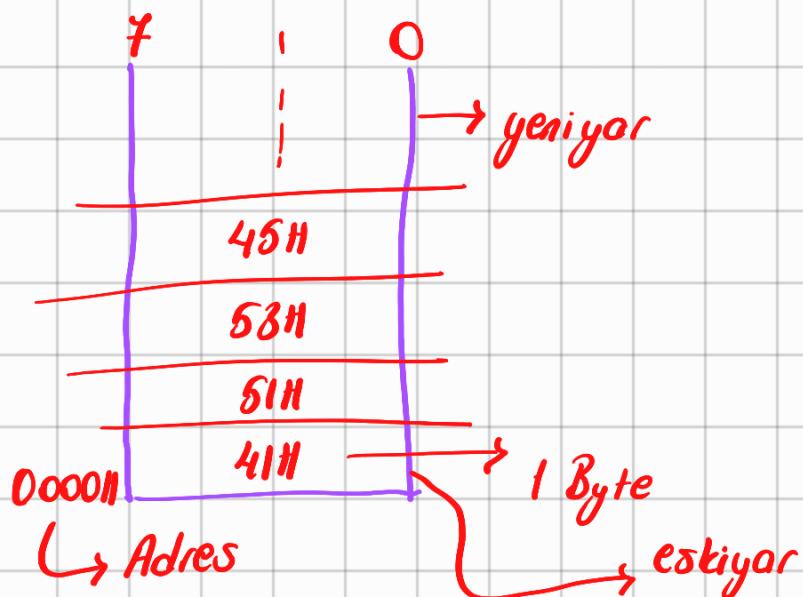
PUSHF

POPF

LAHF

SAHF

t= MovSB (move String Byte)



Birim nedenle istenmeyen su ekliyor

pösterdipi yeden 4 byte olıp yeniyer'e kopyolamak istiyoruz

MOVSW Bu versiyonu da var

[ES:DI] ← [DS:SI]
extra segmentimiz yoksa buası
Data segmenttir DS

Bu itilinin bulun
dugu adresi
diger tarafa
Kopyoluyor

Data segment icindeki SI'nin gösterdigi yeri kopyoluyor
extra segment icindeki DI'nin gösterdigi yere kopyoluyor

CLD
DF → 0 => Adres
articak
JTD => Adres
azalticak

Ortan
adreslere
dogru gidicek

Orton
adreslere
dogru gidicek

LEA SI, eskiyer
LEA DI, yeniyer

MOVSB tek seferde
1 byte'lik işlem

Mor Cx, 4

(CLD)

→ REP moves

Bu islemi

4 kev

tekcademasi.com

Joplor

Adresin
ortmasını
MOSB
yapır
bu doğrultusunda
direction flagin
yönüne bota-
rot yapıyor

yapıyor biz bu
islemi 4 tere
yapıcaz

Diagram illustrating energy levels (H) represented by horizontal red lines. The top set of lines shows levels 45H, 53H, 51H, 41H, and 21H. The bottom set of lines shows levels 45H, 53H, 51H, and 41H. A yellow arrow labeled "yeni yer" points to the 41H level in the top set. A red arrow labeled "eski yer" points to the 41H level in the bottom set.

CMPSB (compare String byte)

Cmpsw holi de vor

$$[DS; SI] - [ES; DI]$$

$$DF \rightarrow 0 \uparrow$$

\swarrow

$$\downarrow 1$$

LEA 51, dizi 1 → Gı kartma işlemi
LEA 51, dizi 2 Cm²'luk

LEA DI, DI212

Mov CX, 10

CLD

Bunlara

REPE

CMPSB

Önek

esit oldugu

deniyor

Dürece devom

et demek

ctrl deli gibi

Gitarmanın Sonucu

boyrotklarda olusuyor

1. den 2.'yi gitirip

1.ye yazmıyoruz

'Assembly 12'

|||||

Burası eşit

'Assembly 128'

değil bundan

Sonra ortık

azapiga düşmüştür

oluyor

JCAJB (Scan String Byte)

JCAJW Bu hali de var

AL - [ES:DI] → Sonuç boyrotklarda

Sıvı adresin

olduğu yerdeki

1 byte

@← 40H (ASCII tablosundaki karşılığı)

LEA DI, email

Mov CX, 12

CLD
MOV AL @
REPNE SCASB

LODSB (load string byte)

LODSW bu hali de var

AL \leftarrow [DS:SI]

Bunun içindeki
1 byte'i alıyor
ve AL'nin içine
yazıyor

STOSB

[ES:DI] \leftarrow AL

CBW (convert byte to word)

AX \leftarrow AL

→ Bu işlem hatasıdır

CBW BL ; BX \leftarrow BL X → Bu işlem doğrudur
CBW V

AL \leftarrow 0001111B AH', sıfırlayarak

$Hx \leftarrow 1111111100011110$ da oynadını yapmış
olmuyoruz

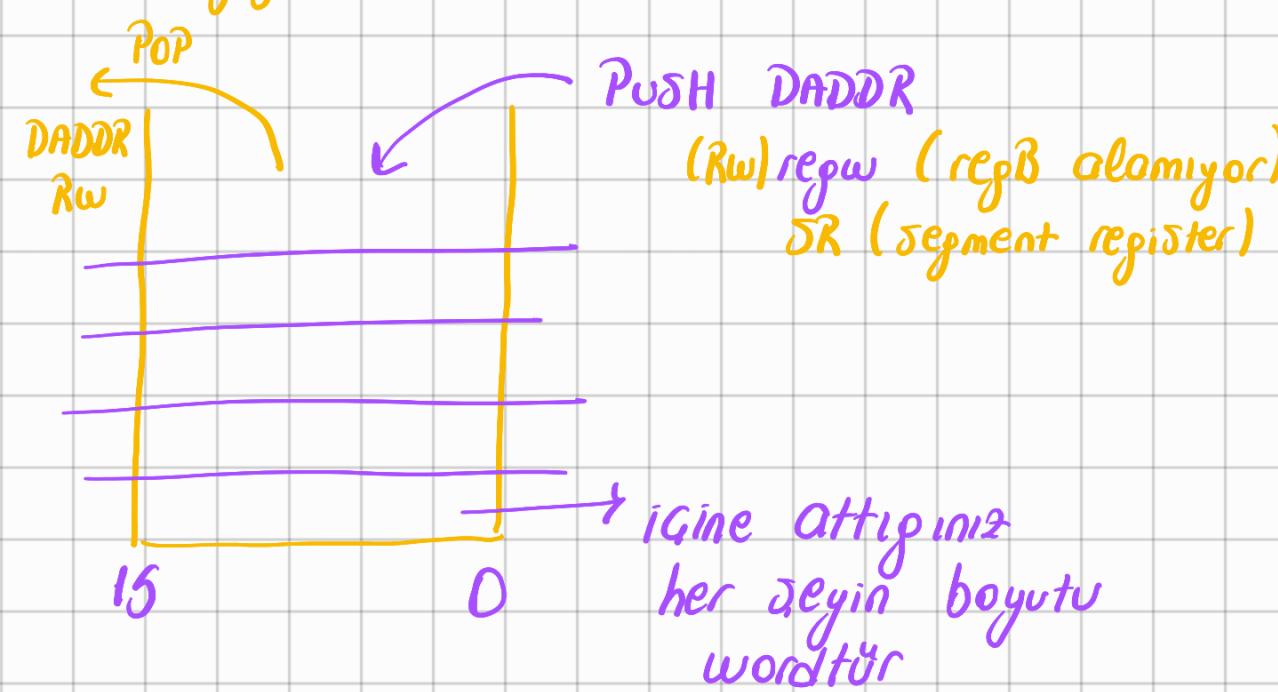
\swarrow
Cbw en onlamlı
biti tescirliyor
deram ettiyor

en onlamlı
biti test komutuya
la bulabiliyoruz

Cwd

$DX:AX \leftarrow AX$

Yığın (stack) Komutları



Mov AX, 2

PUSH AX

JHR AX, 1

PUSH re Pop
1'er operand

PUSH AX

POP BX → 1

ADD BX, 7

PUSH BX

POP DX ← 8

olarlar

Soru

Bizim bir dizimiz var 1 2 3 4 diye
bu bunu 4 3 2 1 diye ters etmek
istiyoruz

dizi[]: 1 2 3 4

4 3 2 1

XOR SI, SI

MOV CX, 4

L1: PUSH dizi[SI]

INC SI

LOOP L1

XOR SI, SI

MOV CX, 4

L2: POP dizi[SI]

INC SI

LOOP L2

PUSHF → PSW

Flags

PSW'nin değerini yani boyrotlarda oluşan değerini stack'in üzerine koymuyor

POPF diyeret o boyrotları stack'ten getebilirsiniz

LAHF → Load AH with Flags

AH'ın içerişine Flagleri alıyor

AH [SF | ZF | ? | AF | ? | PF | ? | CF]

Sadece 5 tane boyroğunu alıyor
bütün boyrotlarını alımıyor

SAHF → Store AH in Flags (tekrar yerine
kaymamak için)

ÖR diper boyrotlara dokunmadan ZF değerinin
Complementini alalım

$$\begin{array}{rcl} 0 & \rightarrow & 1 \\ 1 & \rightarrow & 0 \end{array}$$

Bu da Complementini almak istiyorsunuz

Bu bölümdeki örneğimizde 8 byte'lık bir dosya okuma işlemi yapmayı göstermektedir.

10011010
⊕ 01000000 → Bu değerle → 40H
_____ XOR'ldik

PUSHF → 8 flags

LAHF

XOR AH 40H

POPF

SAHF

Verilen 1 byte'lik bilgileri Jyon ASM Kodunu yazalım

XOR BL, BL → mov BL, 0 } Bunlarda
mov CX, 8 SUB BL, BL } aynı işlemi yapar
mov AH, 55h fakat XOR daha
L1: SHL AH, 1 hızlı yapar
JNC Devam işlem sonucunda 4'ü pürmeyi
INC BL hedefliyoruz

Devam: Loop L1

→ Bu da 2. Gözüm

XOR BL, BL
mov CX, 8
mov AH, 55h
L1: SHL AH, 1

ADC BL, 0
LOOP L1

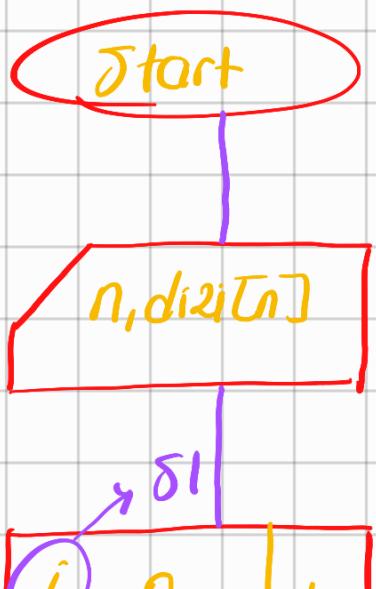
→ Bu da 3. Gözüm

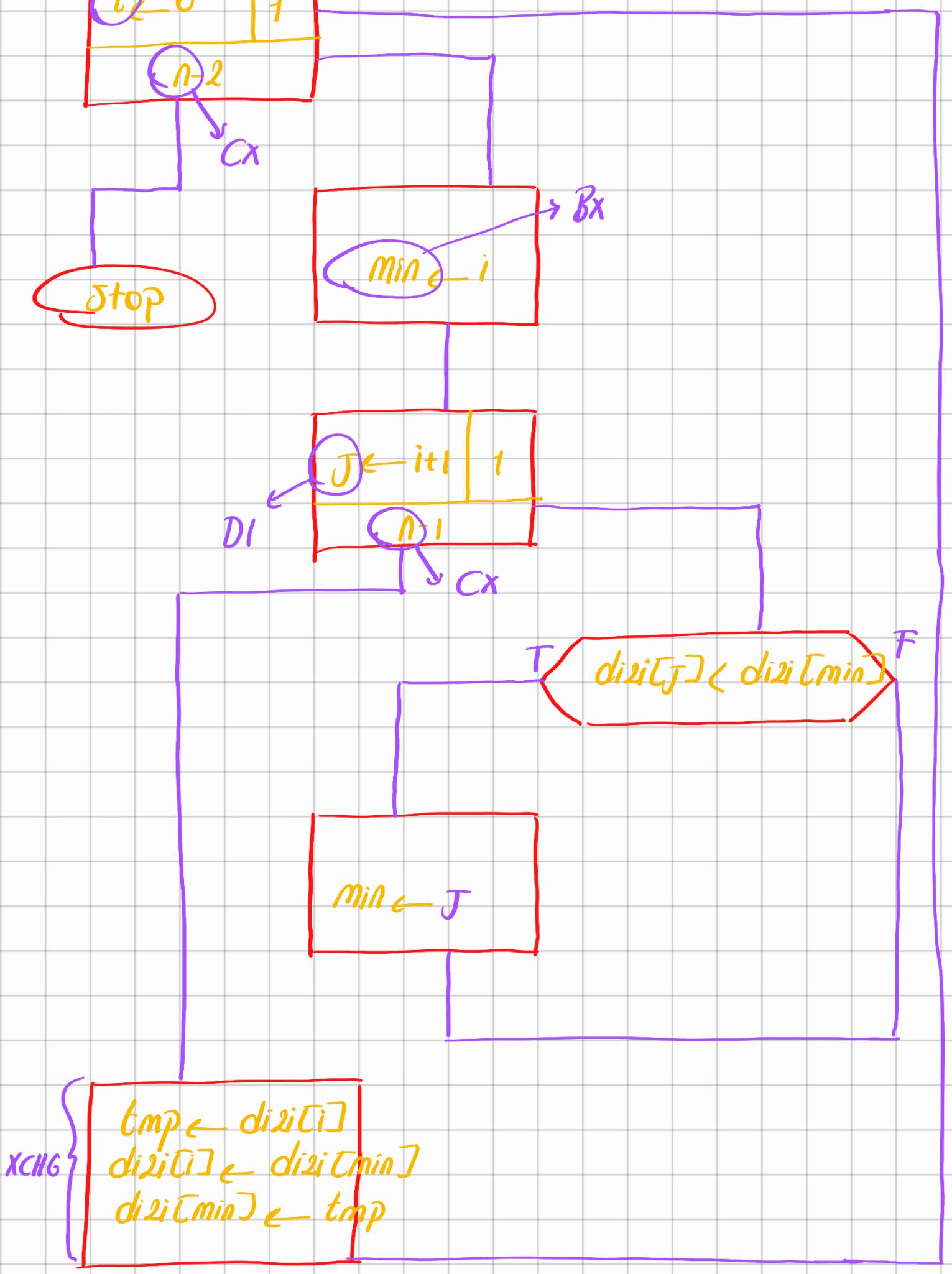
XOR BL, BL
MOV AH, 55h
L1: CMP AH, 0
JE Gitis
SHL AH, 1
ADC BL, 0
JMP L1

Gitis: _____

SORU

Selection Sort





XOR DI, SI ; indisimizi degerini 0'ladıktır

Mov CX, n ; Döngü Sayısı CX'te tutulur

DEC CX ; döngü sayısını 1 azaltırız

dis don; Mov BX, SI ; indisimizi minimuma atonuz

Push CX ; Döngü Sayısını Stack'e attık

Mov DI, SI ; Dis döngü indisini iç döngü indisine atadık

ADD DI, 2

Mov DX, SI

SHR DX, 1

Mov CX, N

Sub CX, DX

İç-dön; Mov AX, dizi[DI]

Cmp AX, dizi[BX]

JGE false col

Mov BX, DI

ADD DI, 2

Loop ig-don

Pop Cx

Mov AX, dizi[SI]

XCHG AX, dizi[BX]

Mov dizi[SI], AX

ADD SI, 2

Loop dis-don

