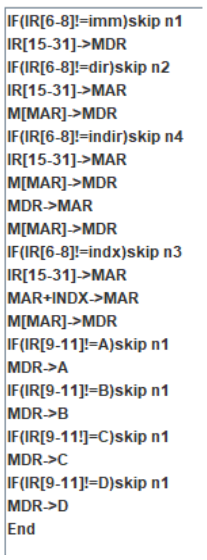
Machine Instructions: -

1. Load

Format: op mode,Dreg,unused addr

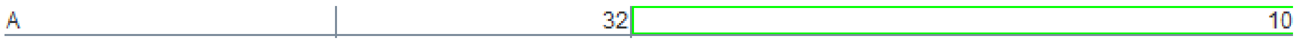
RTL'S :



```
IF(IR[6-8]!=imm)skip n1
IR[15-31]->MDR
IF(IR[6-8]!=dir)skip n2
IR[15-31]->MAR
M[MAR]->MDR
IF(IR[6-8]!=indir)skip n4
IR[15-31]->MAR
M[MAR]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[6-8]!=indx)skip n3
IR[15-31]->MAR
MAR+INDX->MAR
M[MAR]->MDR
IF(IR[9-11]!=A)skip n1
MDR->A
IF(IR[9-11]!=B)skip n1
MDR->B
IF(IR[9-11!]=C)skip n1
MDR->C
IF(IR[9-11]!=D)skip n1
MDR->D
End
```

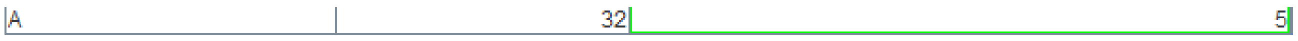
Output:

load imm A,10



```
A                32 | 10
```

load dir A,Var

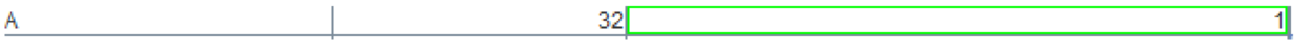


```
A                32 | 5
```

load indir A,ptr

Array: .data 20 4[1,2,3,4,5]

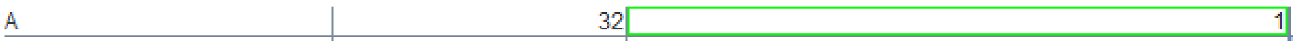
ptr: .data 4 Array



```
A                32 | 1
```

load indx A, Array

Array: .data 20 4[1,2,3,4,5]



```
A                32 | 1
```

2. Loadr

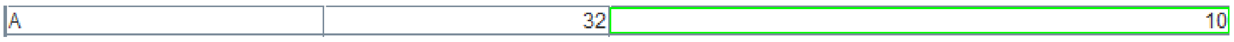
Format: op mode,Dreg,[Sreg],unused2

RTL'S:

```
IF(IR[6-8]!=reg2reg)skip n16
IF(IR[12-14]!=A) skip n1
A->MDR
IF(IR[12-14]!=B) skip n1
B->MDR
IF(IR[12-14]!=C) skip n1
C->MDR
IF(IR[12-14]!=D) skip n1
D->MDR
IF(IR[9-11]!=A)skip n1
MDR->rA
IF(IR[9-11]!=B)skip n1
MDR->rB
IF(IR[9-11!]=C)skip n1
MDR->rC
IF(IR[9-11]!=D)skip n1
MDR->rD
IF(IR[6-8]!=regindir)skip n16
IF(IR[12-14]!=A) skip n1
A->MAR
IF(IR[12-14]!=B) skip n1
B->MAR
IF(IR[12-14]!=C) skip n1
C->MAR
IF(IR[12-14]!=D) skip n1
D->MAR
IF(IR[9-11]!=A)skip n1
M[MAR]->A
IF(IR[9-11]!=B)skip n1
M[MAR]->B
IF(IR[9-11!]=C)skip n1
M[MAR]->C
IF(IR[9-11]!=D)skip n1
M[MAR]->D
End
```

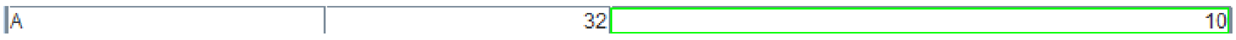
Output:

load imm B,10
loadr reg2reg A,B



A	32	10

load imm B,x
loadr regindir A,[B]
x: .data 4 10



A	32	10

3. Loadrs

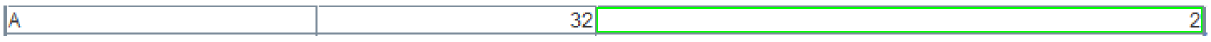
Format: op mode,Dreg,Sreg,addr

RTL'S:

```
IF(IR[6-8]!=imm)skip n1
IR[15-31]->MDR
IF(IR[6-8]!=dir)skip n3
IR[15-31]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[6-8]!=indir)skip n5
IR[15-31]->MDR
MDR->MAR
M[MAR]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[12-14]!=A) skip n1
A->MAR
IF(IR[12-14]!=B) skip n1
B->MAR
IF(IR[12-14]!=C) skip n1
C->MAR
IF(IR[12-14]!=D) skip n1
D->MAR
MAR+MDR->MAR
IF(IR[9-11]!=A)skip n1
M[MAR]->A
IF(IR[9-11]!=B)skip n1
M[MAR]->B
IF(IR[9-11!]=C)skip n1
M[MAR]->C
IF(IR[9-11]!=D)skip n1
M[MAR]->D
End
```

Output:

load imm B,Array
loadrs imm A,B,4 ;[B+10]
Array: .data 20 4[1,2,3,4,5]



A	32	2

4. Store

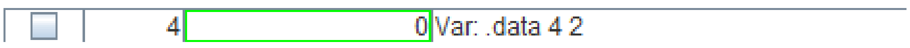
Format: store: op mode,Dreg,unused addr

RTL'S

```
IF(IR[6-8]!=dir)skip n1
IR[15-31]->MAR
IF(IR[6-8]!=indir)skip n3
IR[15-31]->MAR
M[MAR]->MDR
MDR->MAR
IF(IR[6-8]!=indx)skip n2
IR[15-31]->MAR
MAR+INDX->MAR
IF(IR[9-11]!=A)skip n1
A->M[MAR]
IF(IR[9-11]!=B)skip n1
B->M[MAR]
IF(IR[9-11!]=C)skip n1
C->M[MAR]
IF(IR[9-11]!=D)skip n1
D->D[MAR]
End
```

Output:

Store dir A,Var

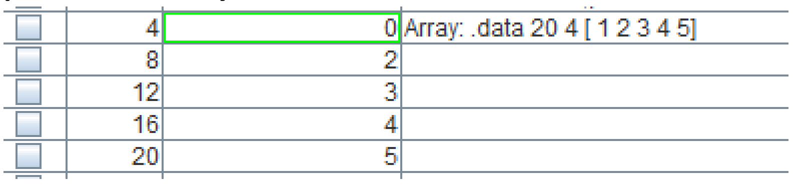


☐	4	0	Var: .data 4 2

Store indir A,ptr

Array: .data 20 4[1,2,3,4,5]

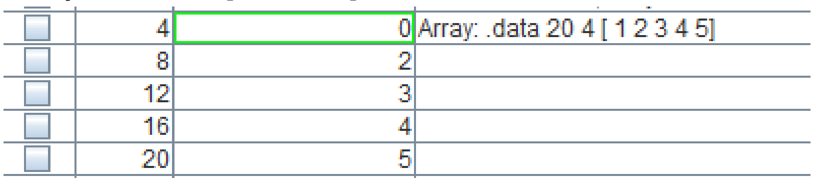
ptr: .data 4 Array



☐	4	0	Array: .data 20 4 [1 2 3 4 5]
☐	8	2	
☐	12	3	
☐	16	4	
☐	20	5	

Store indx A,Array

Array: .data 20 4[1,2,3,4,5]

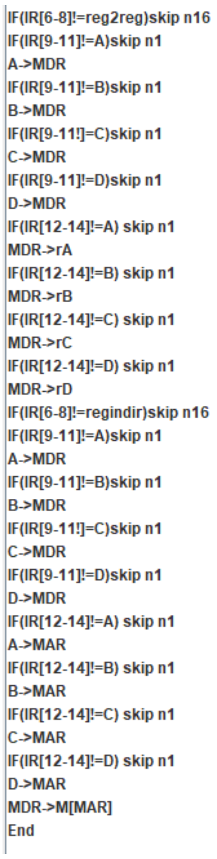


☐	4	0	Array: .data 20 4 [1 2 3 4 5]
☐	8	2	
☐	12	3	
☐	16	4	
☐	20	5	

5. Storer

Format: op mode,Dreg,[Sreg],unused2

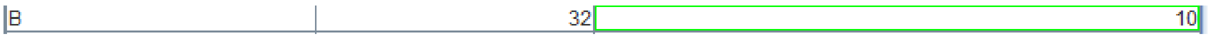
RTL'S:



```
IF(IR[6-8]!=reg2reg)skip n16
IF(IR[9-11]!=A)skip n1
A->MDR
IF(IR[9-11]!=B)skip n1
B->MDR
IF(IR[9-11!]=C)skip n1
C->MDR
IF(IR[9-11]!=D)skip n1
D->MDR
IF(IR[12-14]!=A) skip n1
MDR->rA
IF(IR[12-14]!=B) skip n1
MDR->rB
IF(IR[12-14]!=C) skip n1
MDR->rC
IF(IR[12-14]!=D) skip n1
MDR->rD
IF(IR[6-8]!=regindir)skip n16
IF(IR[9-11]!=A)skip n1
A->MDR
IF(IR[9-11]!=B)skip n1
B->MDR
IF(IR[9-11!]=C)skip n1
C->MDR
IF(IR[9-11]!=D)skip n1
D->MDR
IF(IR[12-14]!=A) skip n1
A->MAR
IF(IR[12-14]!=B) skip n1
B->MAR
IF(IR[12-14]!=C) skip n1
C->MAR
IF(IR[12-14]!=D) skip n1
D->MAR
MDR->M[MAR]
End
```

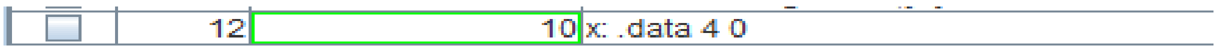
Output:

load imm A,10
storer reg2reg A,B



```
B                32  10
```

load imm A,10
load imm B,x
storer regindir A,[B]
x: .data 4 0



```
    12    10 x: .data 4 0
```

6. Storers

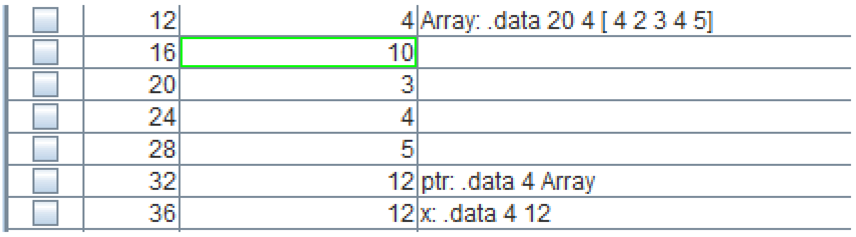
Format: op mode,Dreg,Sreg,addr

RTL'S:

```
IF(IR[6-8]!=imm)skip n1
IR[15-31]->MDR
IF(IR[6-8]!=dir)skip n3
IR[15-31]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[6-8]!=indir)skip n5
IR[15-31]->MDR
MDR->MAR
M[MAR]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[12-14]!=A) skip n1
A->MAR
IF(IR[12-14]!=B) skip n2
B->MAR
IF(IR[12-14]!=C) skip n1
C->MAR
IF(IR[12-14]!=D) skip n1
D->MAR
MAR+MDR->MAR
IF(IR[9-11]!=A)skip n1
A->M[MAR]
IF(IR[9-11]!=B)skip n1
B->M[MAR]
IF(IR[9-11!]=C)skip n1
C->M[MAR]
IF(IR[9-11]!=D)skip n1
D->M[MAR]
End
```

Output:

```
load imm A,10
load dir B,x
storers indir A,B,ptr ;[B+ptr]
Array: .data 20 4[4,2,3,4,5]
ptr: .data 4 Array
x: .data 4 12
```



☐	12	4	Array: .data 20 4 [4 2 3 4 5]
☐	16	10	
☐	20	3	
☐	24	4	
☐	28	5	
☐	32	12	ptr: .data 4 Array
☐	36	12	x: .data 4 12

7. Stop

Format: op unused,unused,unused,unused2

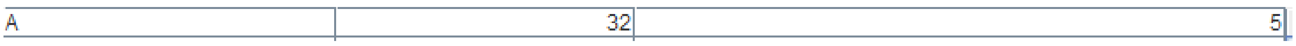
RTL'S

```
set-h bit
End
```

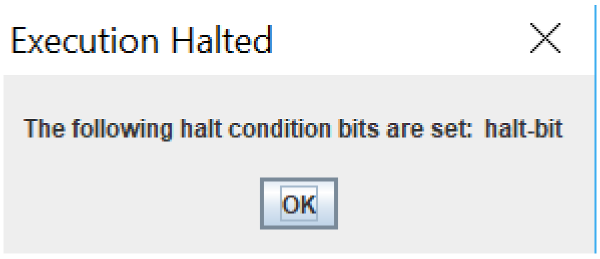
Output:

load imm A,5

stop



A 32 5



Execution Halted

The following halt condition bits are set: halt-bit

8. Jumpe

Format: op unused,unused,unused,addr

RTL'S:

```
IF[Status[1]!=1) skip n1
IR[15-31]->PC
End
```

9. Jumpp

Format: op unused,unused,unused,addr

RTL'S:

```
IF[Status[2]!=1)
IR[15-31]->PC
End
```

10.Jumpn

Format: op unused,unused,unused,addr

RTL'S:

```
IF[Status[2]!=1) skip n1
IR[15-31]->PC
End
```

11.Cmp

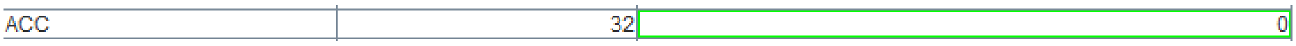
Format: cmp: op mode,Dreg,unused addr

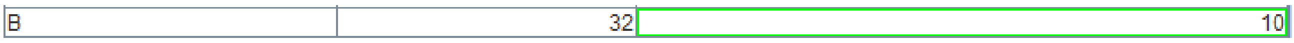
RTL'S:

```
IF(IR[6-8]!=imm)skip n1
IR[15-31]->MDR
IF(IR[6-8]!=dir)skip n2
IR[15-31]->MAR
M[MAR]->MDR
IF(IR[6-8]!=indir)skip n4
IR[15-31]->MAR
M[MAR]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[6-8]!=indx)skip n3
IR[15-31]->MAR
MAR+INDX->MAR
M[MAR]->MDR
IF(IR[9-11]!=A)skip n1
A->ACC
IF(IR[9-11]!=B)skip n1
B->ACC
IF(IR[9-11!]=C)skip n1
C->ACC
IF(IR[9-11]!=D)skip n1
D->ACC
ACC-MDR->MDR
MDR->ACC
IF(ACC!=0) skip n1
set-z bit
IF(ACC[0]!=1) skip n1
set-n bit
End
```

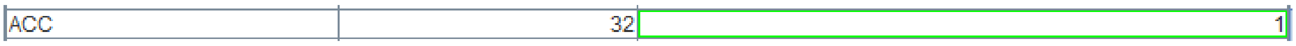
Output:

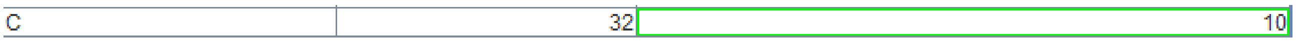
```
load imm A,5    ; When both values are equal
cmp imm A,5
jumpe Equal
Equal:
read B
stop
```


ACC 32 0


B 32 10

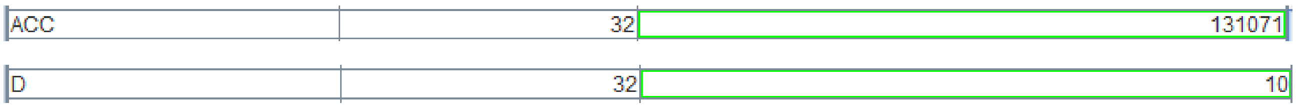
```
load imm A,5  ; When answer is greater than 0
cmp imm A,4
jumpp Pos
Pos:
read C
stop
```


ACC 32 1


C 32 10

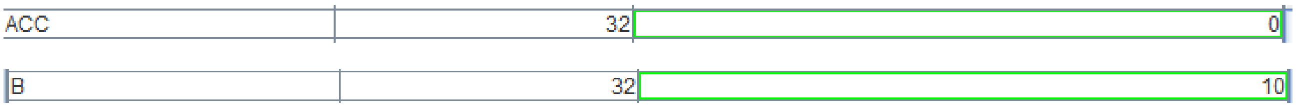
```
load imm A,5
cmp imm A,6
```

```
jumpn Neg
Neg:
read D
stop
```



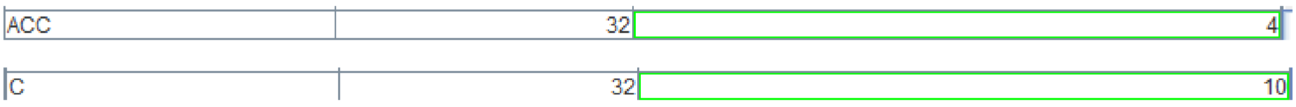
ACC	32	131071
D	32	10

```
load imm A,5  ; When both values are equal
cmp dir A,x
jumpe Equal
Equal:
read B
stop
x:.data 4 5
```



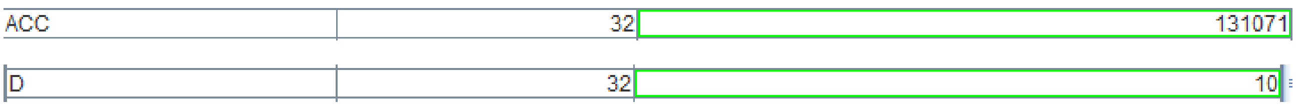
ACC	32	0
B	32	10

```
load imm A,5
cmp indir A,ptr
jumpp Pos
Pos:
read C
stop
Array: .data 20 4[1,2,3,4,5]
ptr: .data 4 Array
```



ACC	32	4
C	32	10

```
load imm A,0
cmp indx A,Array
jumpn Neg
Neg:
read D
stop
Array: .data 20 4[1,2,3,4,5]
```

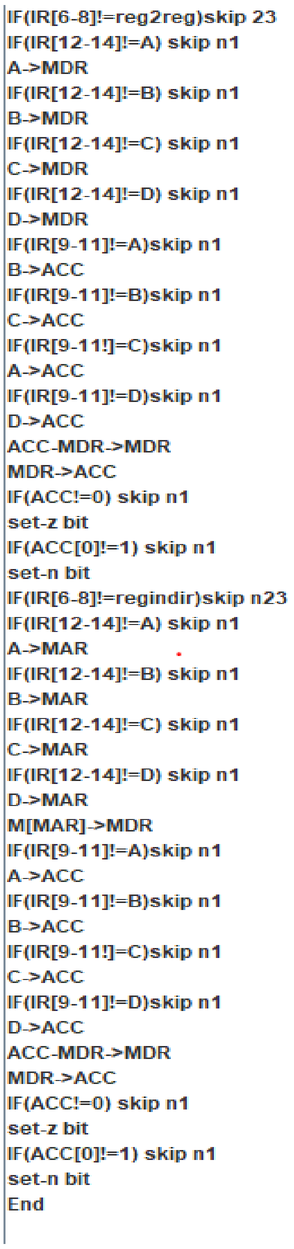


ACC	32	131071
D	32	10

12. Cmpr

Format: op mode,Dreg,[Sreg] unused2

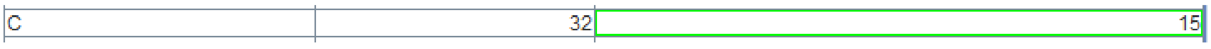
RTL’S:



```
IF(IR[6-8]!=reg2reg)skip 23
IF(IR[12-14]!=A) skip n1
A->MDR
IF(IR[12-14]!=B) skip n1
B->MDR
IF(IR[12-14]!=C) skip n1
C->MDR
IF(IR[12-14]!=D) skip n1
D->MDR
IF(IR[9-11]!=A)skip n1
B->ACC
IF(IR[9-11]!=B)skip n1
C->ACC
IF(IR[9-11!]=C)skip n1
A->ACC
IF(IR[9-11]!=D)skip n1
D->ACC
ACC-MDR->MDR
MDR->ACC
IF(ACC!=0) skip n1
set-z bit
IF(ACC[0]!=1) skip n1
set-n bit
IF(IR[6-8]!=regindir)skip n23
IF(IR[12-14]!=A) skip n1
A->MAR
IF(IR[12-14]!=B) skip n1
B->MAR
IF(IR[12-14]!=C) skip n1
C->MAR
IF(IR[12-14]!=D) skip n1
D->MAR
M[MAR]->MDR
IF(IR[9-11]!=A)skip n1
A->ACC
IF(IR[9-11]!=B)skip n1
B->ACC
IF(IR[9-11!]=C)skip n1
C->ACC
IF(IR[9-11]!=D)skip n1
D->ACC
ACC-MDR->MDR
MDR->ACC
IF(ACC!=0) skip n1
set-z bit
IF(ACC[0]!=1) skip n1
set-n bit
End
```

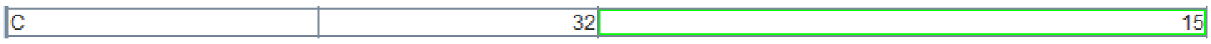
Output:

load imm A,10
load imm B,10
cmpr reg2reg A,B
jumpe Equal
Equal:
read C
stop



C	32	15

```
load imm A,10
load imm B,x
cmpr regindir A,[B]
jumpe Equal
Equal:
read C
stop
x: .data 4 10
```



```
C    32    15
```

13.Cmprs

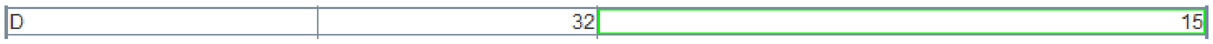
Format: op mode,Dreg,Sreg,addr

RTL'S:

```
IF(IR[6-8]!=imm)skip n1
IR[15-31]->MDR
IF(IR[6-8]!=dir)skip n3
IR[15-31]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[6-8]!=indir)skip n5
IR[15-31]->MDR
MDR->MAR
M[MAR]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[12-14]!=A) skip n1
A->MAR
IF(IR[12-14]!=B) skip n1
B->MAR
IF(IR[12-14]!=C) skip n1
C->MAR
IF(IR[12-14]!=D) skip n1
D->MAR
MAR+MDR->MAR
M[MAR]->MDR
IF(IR[9-11]!=A)skip n1
A->ACC
IF(IR[9-11]!=B)skip n1
B->ACC
IF(IR[9-11!]=C)skip n1
C->ACC
IF(IR[9-11]!=D)skip n1
D->ACC
ACC-MDR->MDR
MDR->ACC
IF(ACC!=0) skip n1
set-z bit
IF(ACC[0]!=1) skip n1
set-n bit
End
```

Output:

```
load imm A,10
load imm B,Array
cmprs dir A,B,x ;[B+X]
jumpe Equal
Equal:
read D
stop
x:.data 4 4
Array: .data 20 4[10,2,3,4,5]
```

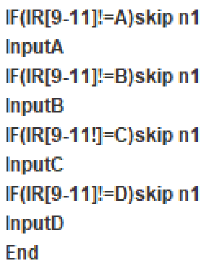


```
D    32    15
```

14. Read

Format: op unused,Dreg,unused,unused2

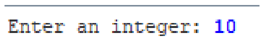
RTL'S:



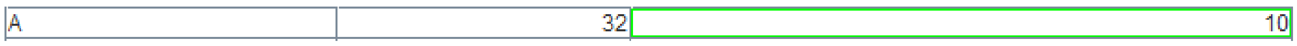
```
IF(IR[9-11]!=A)skip n1
InputA
IF(IR[9-11]!=B)skip n1
InputB
IF(IR[9-11!]=C)skip n1
InputC
IF(IR[9-11]!=D)skip n1
InputD
End
```

Output:

read A



```
Enter an integer: 10
```

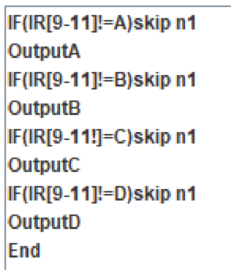


```
A    32    10
```

15.Write

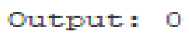
Format: op unused,Dreg,unused,unused2

RTL'S:



```
IF(IR[9-11]!=A)skip n1
OutputA
IF(IR[9-11]!=B)skip n1
OutputB
IF(IR[9-11!]=C)skip n1
OutputC
IF(IR[9-11]!=D)skip n1
OutputD
End
```

Output:



```
Output: 0
```

16.Dec

Format: op unused,Dreg,unused,unused2

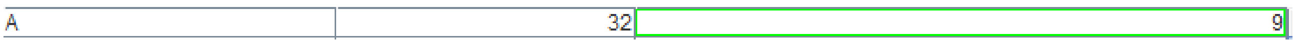
RTL'S:

```
IF(IR[9-11]!=A)skip n2
A-1
A->MDR
IF(IR[9-11]!=B)skip n2
B-1
B->MDR
IF(IR[9-11!]=C)skip n2
C-1
C->MDR
IF(IR[9-11]!=D)skip n2
D-1
D->MDR
IF(MDR!=0) skip n1
set-z bit
IF(MDR[0]!=1) skip n1
set-n bit
End
```

Output:

load imm A,10

Dec A



A	32	9

17. Add

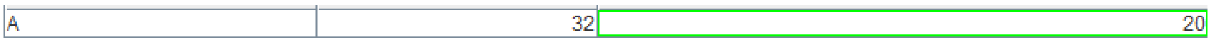
Format: op mode,Dreg,unused addr

RTL'S:

```
0->MDR
IF(IR[6-8]!=imm)skip n1
IR[15-31]->MDR
IF(IR[6-8]!=dir)skip n2
IR[15-31]->MAR
M[MAR]->MDR
IF(IR[6-8]!=indir)skip n4
IR[15-31]->MAR
M[MAR]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[6-8]!=indx)skip n3
IR[15-31]->MAR
MAR+INDX->MAR
M[MAR]->MDR
IF(IR[9-11]!=A)skip n1
A+MDR->A
IF(IR[9-11]!=B)skip n1
B+MDR->B
IF(IR[9-11!]=C)skip n1
C+MDR->C
IF(IR[9-11]!=D)skip n1
D+MDR->D
End
```

Output:

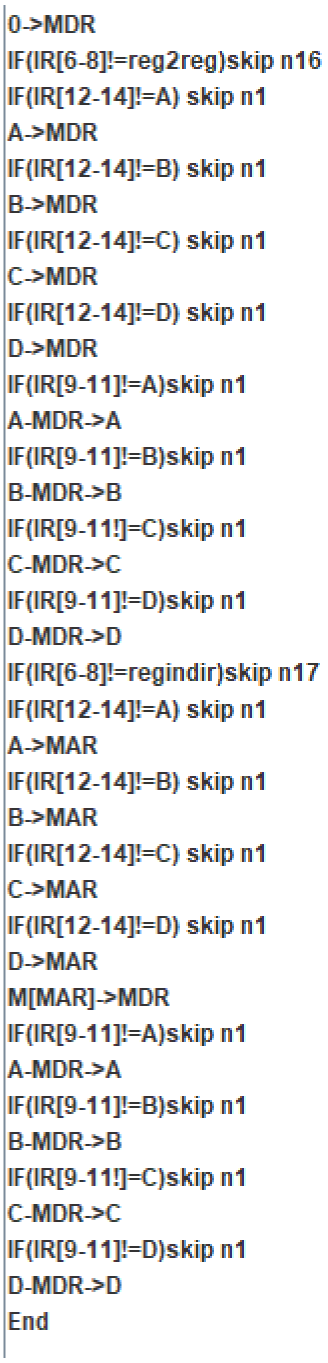
load imm A,10
add dir, A,x
x:.data 4 10



A	32	20

18. Subr

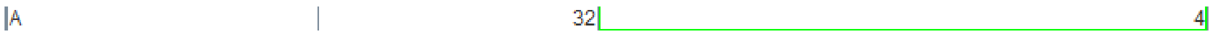
Format: op mode,Dreg,[Sreg],unused2
RTL'S:



```
0->MDR
IF(IR[6-8]!=reg2reg)skip n16
IF(IR[12-14]!=A) skip n1
A->MDR
IF(IR[12-14]!=B) skip n1
B->MDR
IF(IR[12-14]!=C) skip n1
C->MDR
IF(IR[12-14]!=D) skip n1
D->MDR
IF(IR[9-11]!=A)skip n1
A-MDR->A
IF(IR[9-11]!=B)skip n1
B-MDR->B
IF(IR[9-11!]=C)skip n1
C-MDR->C
IF(IR[9-11]!=D)skip n1
D-MDR->D
IF(IR[6-8]!=regindir)skip n17
IF(IR[12-14]!=A) skip n1
A->MAR
IF(IR[12-14]!=B) skip n1
B->MAR
IF(IR[12-14]!=C) skip n1
C->MAR
IF(IR[12-14]!=D) skip n1
D->MAR
M[MAR]->MDR
IF(IR[9-11]!=A)skip n1
A-MDR->A
IF(IR[9-11]!=B)skip n1
B-MDR->B
IF(IR[9-11!]=C)skip n1
C-MDR->C
IF(IR[9-11]!=D)skip n1
D-MDR->D
End
```

Output:

```
load imm A,10
load imm B,x
subr regindir A,[B]
x: .data 4 6
```


A | 32 | 4

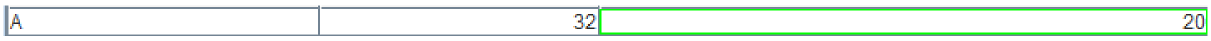
19. Mulrs

Format: op mode,Dreg,[Sreg],unused2
RTL'S:

```
IF(IR[6-8]!=imm)skip n1
IR[15-31]->MDR
IF(IR[6-8]!=dir)skip n3
IR[15-31]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[6-8]!=indir)skip n5
IR[15-31]->MDR
MDR->MAR
M[MAR]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[12-14]!=A) skip n1
A->MAR
IF(IR[12-14]!=B) skip n1
B->MAR
IF(IR[12-14]!=C) skip n1
C->MAR
IF(IR[12-14]!=D) skip n1
D->MAR
MAR+MDR->MAR
M[MAR]->MDR
IF(IR[9-11]!=A)skip n1
A*MDR->A
IF(IR[9-11]!=B)skip n1
B*MDR->B
IF(IR[9-11!]=C)skip n1
C*MDR->C
IF(IR[9-11]!=D)skip n1
D*MDR->D
End
```

Output:

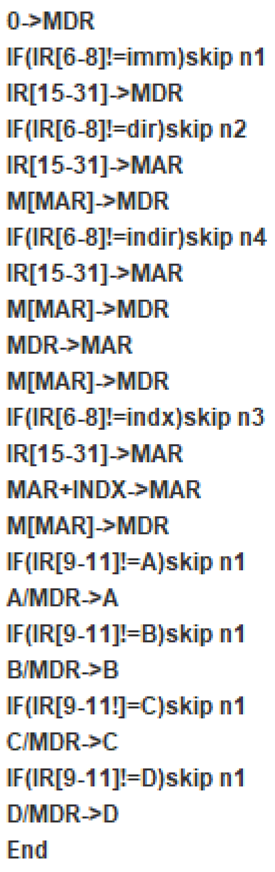
```
load imm A,10
load imm B,Array
mulrs dir A,B,x ;[B+x]
Array: .data 20 4[1,2,3,4,5]
x:.data 4 4
```


A | 32 | 20

20.Div

Format: op mode,Dreg,unused addr

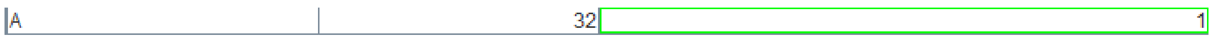
RTL'S:



```
0->MDR
IF(IR[6-8]!=imm)skip n1
IR[15-31]->MDR
IF(IR[6-8]!=dir)skip n2
IR[15-31]->MAR
M[MAR]->MDR
IF(IR[6-8]!=indir)skip n4
IR[15-31]->MAR
M[MAR]->MDR
MDR->MAR
M[MAR]->MDR
IF(IR[6-8]!=indx)skip n3
IR[15-31]->MAR
MAR+INDX->MAR
M[MAR]->MDR
IF(IR[9-11]!=A)skip n1
A/MDR->A
IF(IR[9-11]!=B)skip n1
B/MDR->B
IF(IR[9-11!]=C)skip n1
C/MDR->C
IF(IR[9-11]!=D)skip n1
D/MDR->D
End
```

Output:

load imm A,10
div indir A,ptr
ptr:.data 4 Array
Array: .data 20 4[10,2,3,4,5]



```
A                32  1
```