Microprocessor Lab Exam Report

Rwithik Manoj, Roll No: 53, TVE17CS054

Problem

Identify prime numbers from a given array.

Example: Input: 1,2,3,4,5,6,7,8,9,10
output:1,2,3,5,7

Code

```
; Rwithik Manoj
; Roll No: 53
; TVE17CS054
; Identify prime numbers in an array

DATA SEGMENT
    ARR DB 1,2,3,4,5,6,7,8,9,10
    RESULT DB 20 DUP 10 (?)
    N DB 10
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA
    START:
      MOV AX,DATA
      MOV DS,AX
      LEA SI,ARR            ; Move the address of the first element of the array to SI
      LEA DI,RESULT         ; Set DI to the starting address of the result
      MOV CL,N              ; Set CL to the number of elements, N

      MOV DX,0000H
      MOV AH,00H
      LP0:
      DEC CL                ; Decrement CL
      MOV AL,[SI]           ; Move the next number to be checked to AL
      MOV BL,02H            ; Set BL to 2
      MOV BH,00H            ; Clear BH. BH store the number of divisors greater than 1
      CMP AL,02H            ; If the number is less than 2, goto lebel PRIME
      JLE PRIME

      LP1:
        DIV BL              ; Divide AL by BL
        CMP AH,00H          ; Compare the remainder
        JNE NEXT            ; If remainder is not zero, goto lable NEXT
        INC BH              ; Else increment BH
        NEXT:
          CMP BH,01H        ; If BH >= 1, then the number is not prime.
          JE NONPRIME
          INC BL            ; Otherwise, increment BL to the next number
          MOV DX,0000H
          MOV AH,00H
          MOV AL,[SI]       ; Reset AL to the original number, AL was overwritten during div
          CMP BL,AL         ; If BL is less than AL, do the inner loop again.
          JNE LP1
       PRIME:
          MOV [DI],AL       ; If the number is prime, store it in the results array, and inc
          INC DI
       NONPRIME:
         INC SI             ; Increment SI either way
         CMP CL,0
         JNE LP0            ; If CL is greater than zero, do the outer loop again
   MOV AH,4CH
   INT 21H
CODE ENDS
END START
```

Input 1

1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Output 1

1, 2, 3, 5, 7

Input 2

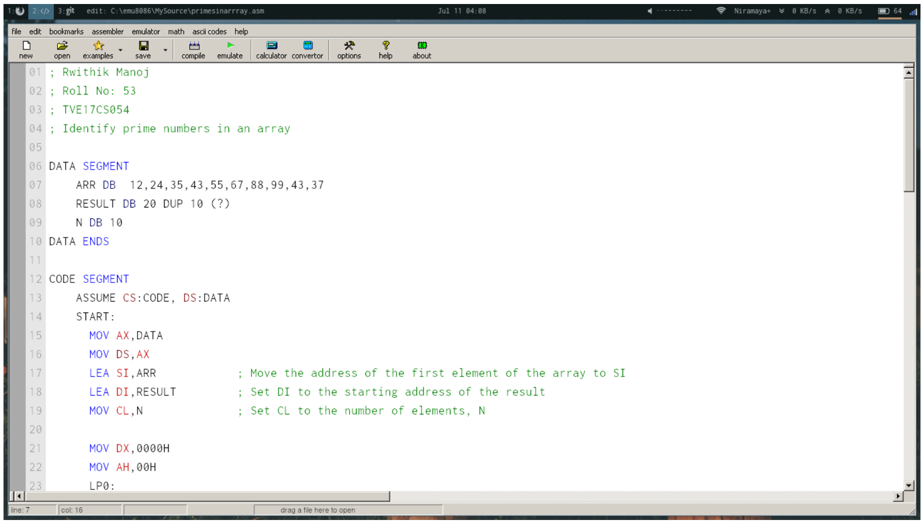
12, 24, 35, 43, 55, 67, 88, 99, 44, 37

Output 2

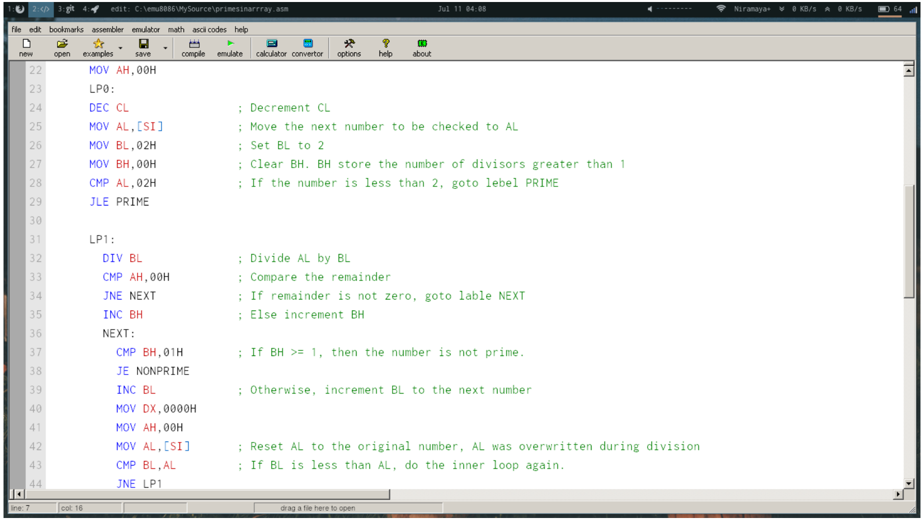
43, 67, 37

Screenshots

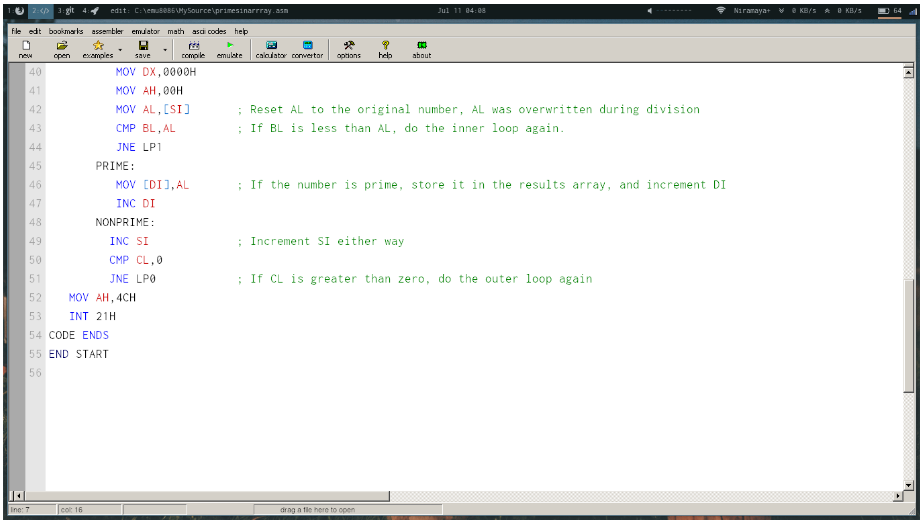
Code


1: 2:</> 3:git edit: C:\emu8086\MySource\primesinarrray.asm Jul 11 04:08 Niramaya+ 0 KB/s 0 KB/s 64
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
line: 7 col: 16 drag a file here to open

```
; Rwithik Manoj
; Roll No: 53
; TVE17CS054
; Identify prime numbers in an array

DATA SEGMENT
    ARR DB  12,24,35,43,55,67,88,99,43,37
    RESULT DB 20 DUP 10 (?)
    N DB 10
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA
    START:
      MOV AX,DATA
      MOV DS,AX
      LEA SI,ARR              ; Move the address of the first element of the array to SI
      LEA DI,RESULT           ; Set DI to the starting address of the result
      MOV CL,N                ; Set CL to the number of elements, N

      MOV DX,0000H
      MOV AH,00H
      LP0:
```

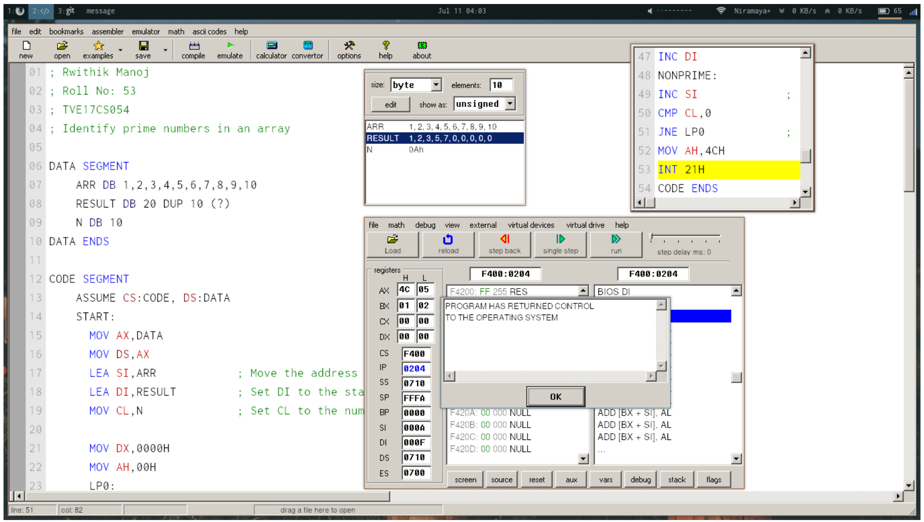

1: 2:</> 3:git 4: edit: C:\emu8086\MySource\primesinarrray.asm Jul 11 04:08 Niramaya+ 0 KB/s 0 KB/s 64
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
line: 7 col: 16 drag a file here to open

```
      MOV AH,00H
      LP0:
      DEC CL                  ; Decrement CL
      MOV AL,[SI]             ; Move the next number to be checked to AL
      MOV BL,02H              ; Set BL to 2
      MOV BH,00H              ; Clear BH. BH store the number of divisors greater than 1
      CMP AL,02H              ; If the number is less than 2, goto lebel PRIME
      JLE PRIME

      LP1:
        DIV BL                ; Divide AL by BL
        CMP AH,00H            ; Compare the remainder
        JNE NEXT              ; If remainder is not zero, goto lable NEXT
        INC BH                ; Else increment BH
        NEXT:
          CMP BH,01H          ; If BH >= 1, then the number is not prime.
          JE NONPRIME
          INC BL              ; Otherwise, increment BL to the next number
          MOV DX,0000H
          MOV AH,00H
          MOV AL,[SI]         ; Reset AL to the original number, AL was overwritten during division
          CMP BL,AL           ; If BL is less than AL, do the inner loop again.
          JNE LP1
```



```
          MOV DX,0000H
          MOV AH,00H
          MOV AL,[SI]                 ; Reset AL to the original number, AL was overwritten during division
          CMP BL,AL                   ; If BL is less than AL, do the inner loop again.
          JNE LP1
       PRIME:
          MOV [DI],AL                 ; If the number is prime, store it in the results array, and increment DI
          INC DI
       NONPRIME:
         INC SI                       ; Increment SI either way
         CMP CL,0
         JNE LP0                      ; If CL is greater than zero, do the outer loop again
   MOV AH,4CH
   INT 21H
CODE ENDS
END START
```

Inputs/Outputs


1: 2:</> 3:git message
Jul 11 04:03
Niramaya+ 0 KB/s 0 KB/s
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
file math debug view external virtual devices virtual drive help
Load reload step back single step run
step delay ms: 0
registers
screen source reset aux vars debug stack flags
line: 51 col: 82
drag a file here to open

```
; Rwithik Manoj
; Roll No: 53
; TVE17CS054
; Identify prime numbers in an array

DATA SEGMENT
    ARR DB 1,2,3,4,5,6,7,8,9,10
    RESULT DB 20 DUP 10 (?)
    N DB 10
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA
    START:
      MOV AX,DATA
      MOV DS,AX
      LEA SI,ARR             ; Move the address
      LEA DI,RESULT          ; Set DI to the sta
      MOV CL,N               ; Set CL to the num

      MOV DX,0000H
      MOV AH,00H
      LP0:
```

```
INC DI
NONPRIME:
INC SI              ;
CMP CL,0
JNE LP0             ;
MOV AH,4CH
INT 21H
CODE ENDS
```

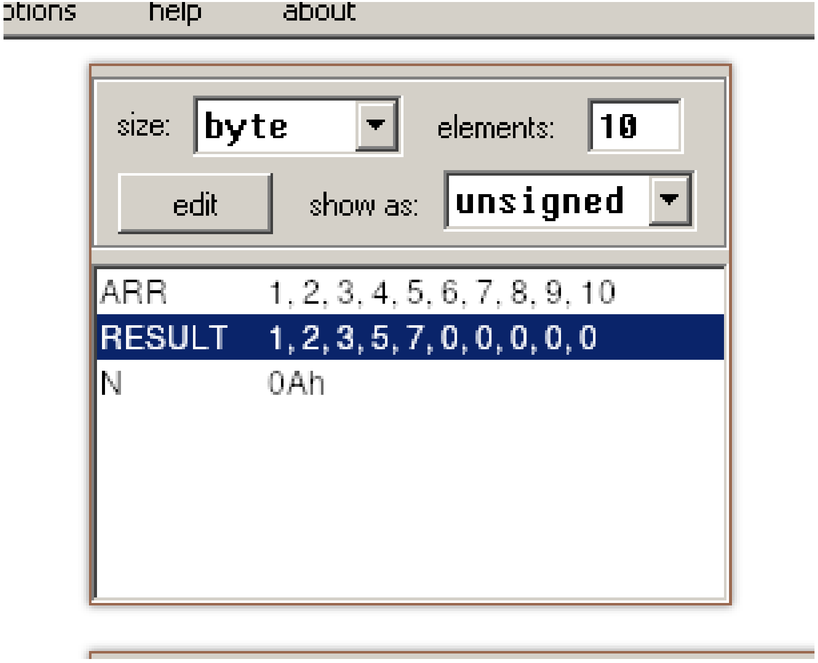
size: byte elements: 10
edit show as: unsigned

```
ARR       1, 2, 3, 4, 5, 6, 7, 8, 9, 10
RESULT    1,2,3,5,7,0,0,0,0,0
N         0Ah
```

	H	L
AX	4C	05
BX	01	02
CX	00	00
DX	00	00
CS	F400	
IP	0204	
SS	0710	
SP	FFFA	
BP	0000	
SI	000A	
DI	000F	
DS	0710	
ES	0700	

```
F400:0204                      F400:0204
F4200: FF 255 RES              BIOS DI
F420A: 00 000 NULL             ADD [BX + SI], AL
F420B: 00 000 NULL             ADD [BX + SI], AL
F420C: 00 000 NULL             ADD [BX + SI], AL
F420D: 00 000 NULL             ...
```

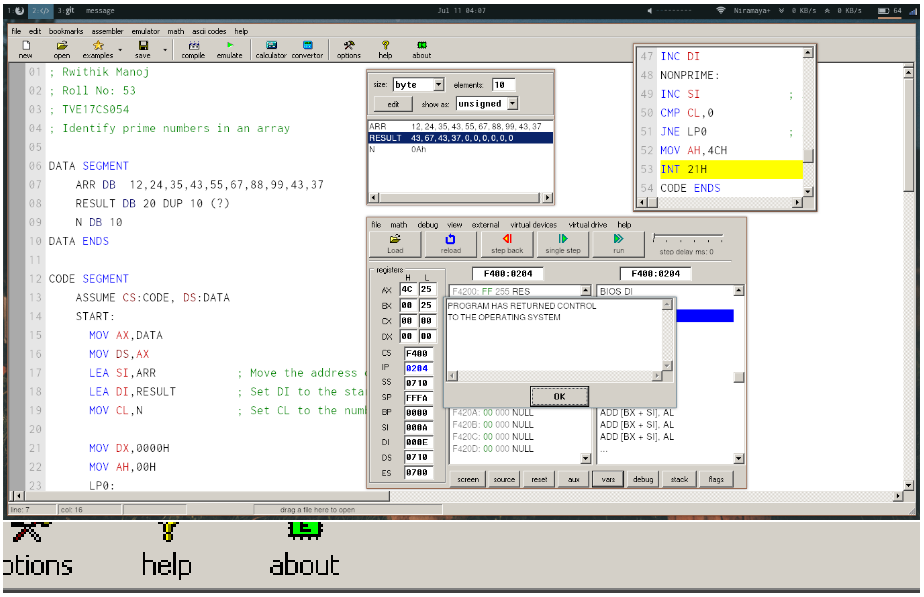
```
PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM
```



size: byte elements: 10

edit show as: unsigned

```
ARR       1, 2, 3, 4, 5, 6, 7, 8, 9, 10
RESULT    1,2,3,5,7,0,0,0,0,0
N         0Ah
```


file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
file math debug view external virtual devices virtual drive help
Load reload step back single step run
step delay ms: 0
screen source reset aux vars debug stack flags
line: 7
col: 16
drag a file here to open
ptions help about

```
; Rwithik Manoj
; Roll No: 53
; TVE17CS054
; Identify prime numbers in an array

DATA SEGMENT
    ARR DB  12,24,35,43,55,67,88,99,43,37
    RESULT DB 20 DUP 10 (?)
    N DB 10
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA
    START:
      MOV AX,DATA
      MOV DS,AX
      LEA SI,ARR              ; Move the address 
      LEA DI,RESULT           ; Set DI to the sta
      MOV CL,N                ; Set CL to the numb

      MOV DX,0000H
      MOV AH,00H
      LP0:
```

```
INC DI
NONPRIME:
INC SI                       ;
CMP CL,0
JNE LP0                      ;
MOV AH,4CH
INT 21H
CODE ENDS
```

size: byte elements: 10
show as: unsigned

```
ARR      12, 24, 35, 43, 55, 67, 88, 99, 43, 37
RESULT   43, 67, 43, 37, 0, 0, 0, 0, 0, 0
N        0Ah
```

registers

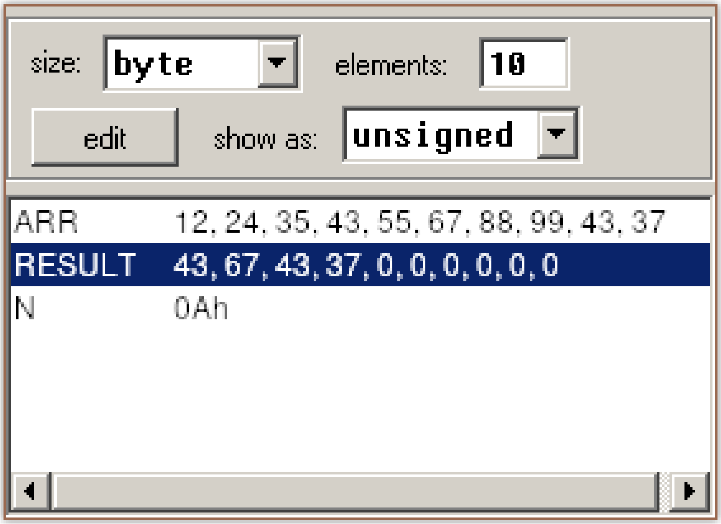
	H	L
AX	4C	25
BX	00	25
CX	00	00
DX	00	00
CS	F400	
IP	0204	
SS	0710	
SP	FFFA	
BP	0000	
SI	000A	
DI	000E	
DS	0710	
ES	0700	

F400:0204 F400:0204

```
F4200: FF 255 RES        BIOS DI
F420A: 00 000 NULL       ADD [BX + SI], AL
F420B: 00 000 NULL       ADD [BX + SI], AL
F420C: 00 000 NULL       ADD [BX + SI], AL
F420D: 00 000 NULL       ...
```

PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

OK



size: byte elements: 10
show as: unsigned

```
ARR      12, 24, 35, 43, 55, 67, 88, 99, 43, 37
RESULT   43, 67, 43, 37, 0, 0, 0, 0, 0, 0
N        0Ah
```