
Q1. In the procedure called "Q1", write assembly code that reads 3 inputs from the user:

1) A number N, where $1 \leq N \leq 10$

2) N intervals, where each interval consists of 2 numbers, a start and end number

3) A number M

Find the number of intervals that the input number M exists in.

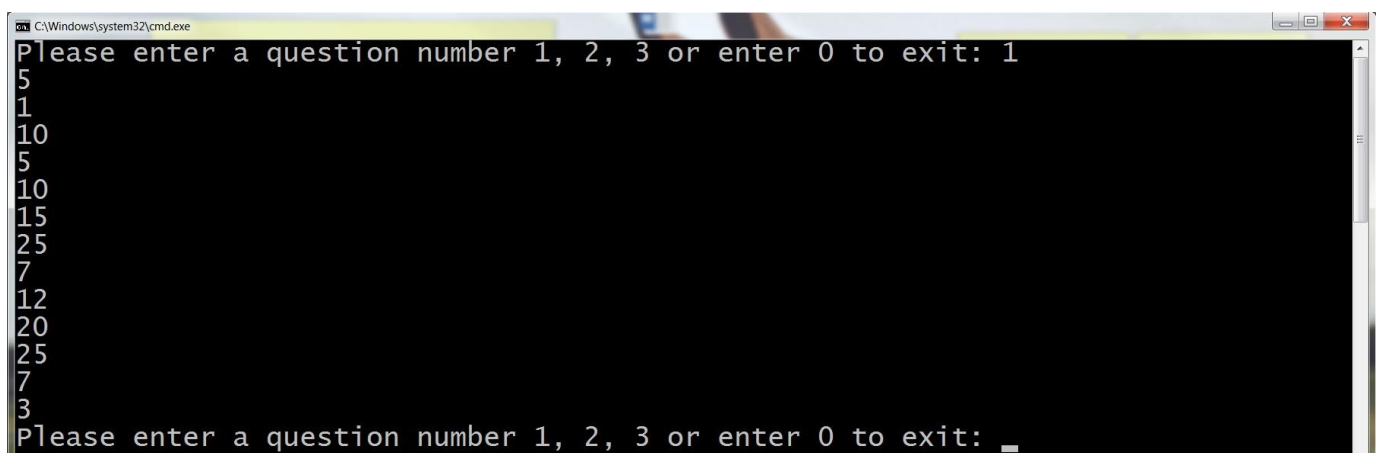
Sample Input:

```
5      ; N
1      ; Interval 1
10
5      ; Interval 2
10
15     ; Interval 3
25
7      ; Interval 4
12
20     ; Interval 5
25
7      ; M
```

Sample Output:

```
3      ; The number 7 exists in 3 of the 5 input intervals: [1,10], [5,10],
[7,12]
```

Sample Run:



```
C:\Windows\system32\cmd.exe
Please enter a question number 1, 2, 3 or enter 0 to exit: 1
5
1
10
5
10
15
25
7
12
20
25
7
3
Please enter a question number 1, 2, 3 or enter 0 to exit: _
```

Figure 1 question-1 sample run

Q2. In the procedure called “Q2”, write assembly code that prints the maximum 2 numbers and the minimum two numbers in an input array. You should print the maximum and the minimum numbers sorted in ascending order. Array type is DWORD, maximum input array length is 10, the max input element is 1000 and the minimum input element is -1000. Use READINT AND WRITEINT methods (Refer to Hints section).

Sample input:

0 -1 -8 3 7 4 255 -90 -10 9

14 8 5 0 -2 -3 -4 11 20 80

Sample output:

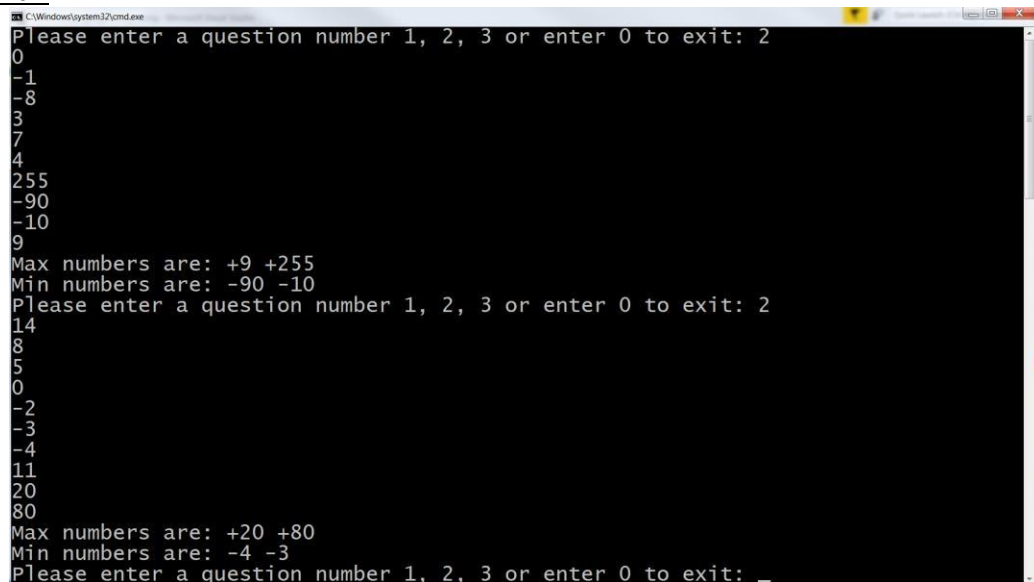
Max numbers are: +9 +255

Min numbers are: -90 -10

Max numbers are: +20 +80

Min numbers are: -4 -3

Sample Run



```
C:\Windows\system32\cmd.exe
Please enter a question number 1, 2, 3 or enter 0 to exit: 2
0
-1
-8
3
7
4
255
-90
-10
9
Max numbers are: +9 +255
Min numbers are: -90 -10
Please enter a question number 1, 2, 3 or enter 0 to exit: 2
14
8
5
0
-2
-3
-4
11
20
80
Max numbers are: +20 +80
Min numbers are: -4 -3
Please enter a question number 1, 2, 3 or enter 0 to exit: _
```

Figure 2 question-2 sample run

Q3. In the procedure called “Q3”, write an assembly code that reads an array size N (where $1 \leq N \leq 10$), then reads N positive numbers. Find the total sum after performing the bit wise OR operation on all the sub arrays of the given array.

Simple Approach: A simple approach is to find the bitwise OR of each subarray of the given array using two nested loops, and then find the total sum. Time complexity of this approach will be $O(N^2)$.

Constraints:

1. Array size N (where $1 \leq N \leq 10$).

1	2	3	4	5
---	---	---	---	---

Sub array	Bitwise OR
{1}	1
{1,2}	3
{1,2,3}	3
{1,2,3,4}	7
{1,2,3,4,5}	7
{2}	2
{2,3}	3
{2,3,4}	7
{2,3,4,5}	7
{3}	3
{3,4}	7
{3,4,5}	7
{4}	4
{4,5}	5
{5}	5
Total	71

Figure 3 explanation on the array {1, 2, 3, 4, 5}

Sample Input:

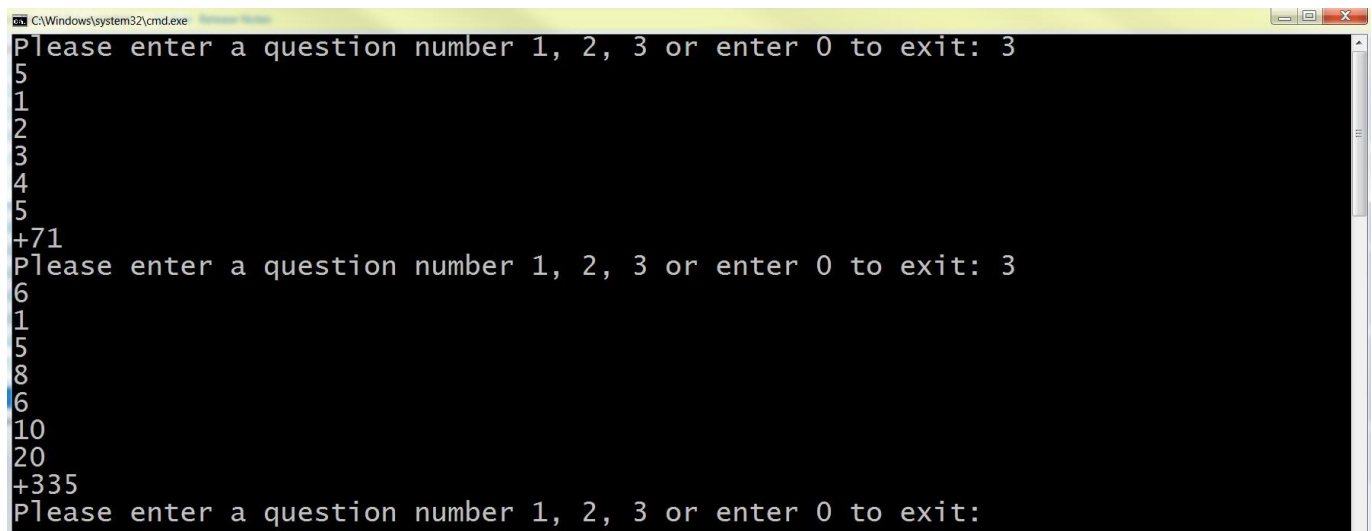
5
1
2
3
4
5

6
1
5
8
6
10
20

Sample Output:

+71
+335

Sample Run



```
C:\Windows\system32\cmd.exe
Please enter a question number 1, 2, 3 or enter 0 to exit: 3
5
1
2
3
4
5
+71
Please enter a question number 1, 2, 3 or enter 0 to exit: 3
6
1
5
8
6
10
20
+335
Please enter a question number 1, 2, 3 or enter 0 to exit:
```

Figure 4 question-3 sample run

Hints:

- `WriteInt` is an Irvine function that prints an integer value that must be stored in EAX register (number's sign is printed).
- `ReadInt` is an Irvine function that reads an integer from the keyboard and stores it in EAX register (the input integer is signed).
- `WriteDec` is an Irvine function that prints an integer value that must be stored in EAX register (number's sign is not printed).
- `ReadDec` is an Irvine function that reads an integer from the keyboard and stores it in EAX register (the input integer is not signed).
- `WriteChar` is an Irvine function that prints a character that must be stored in AL register.
- `ReadString` is an Irvine function that reads a string from the keyboard, stopping when the user presses the Enter key. Pass the offset of a buffer in EDI and set ECX to the maximum number of characters the user can enter. The procedure returns the count of the number of characters typed by the user in EAX.
- `WriteString` is an Irvine function that writes a string to the console. Pass the offset of a buffer in EDI.
- `ReadHex` used to read a hexadecimal value from the user. The value after the read is stored in EAX register.
- `WriteHex` used to write a hexadecimal value to the screen. The value to be displayed is stored in EAX register before calling this procedure.
- `ReadChar` is used to read a char from the console. The value read from the console is placed in "al" register.
- More about these functions and similar ones can be found in section 5.3 of the book.