|  |  |  |  |
| --- | --- | --- | --- |
| NAME: | Roy Lorenz Cardenas, Joshua Sarcol | DATE: | 11/25/ |

PRACTICE EXERCISE # 8.1

# LE8\_11 Days of the Week

Write a function that sets up an array called days, which contains pointers to the names of the days of the week and return the name of the day from the given day. The function accepts the given day.

Example output:

Enter day: 1 Day of the week: Monday

Enter day: 7 Day of the week: Sunday

Enter day: 9 Day of the week: INVALID

## Pseudocode: LE8\_11 Days of the Week

main()

START

1. INITIALIZE input as integer
2. PROMPT and GET input
3. CALL module, result = days(input)
4. PRINT result

STOP

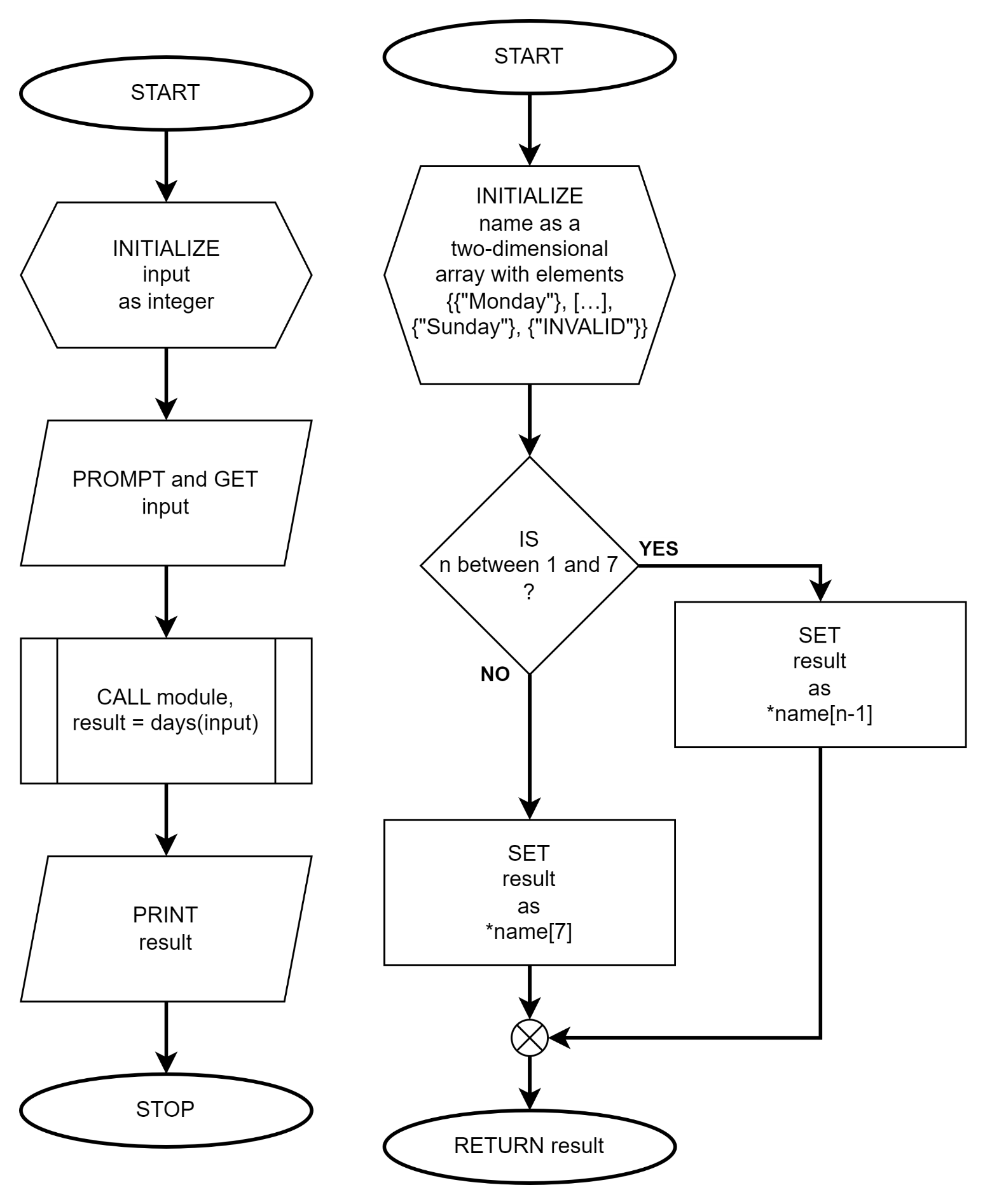
days(n)

START

1. INITIALIZE name as a two-dimensional array with elements {{"Monday"}, {“Tuesday”}, {“Wednesday”}, {“Thursday”}, {“Friday”}, {“Saturday”}, {"Sunday"}, {"INVALID"}}
2. IF n is between 1 to 7
   1. SET result as \*name[n-1]
3. ELSE
4. SET result as \*name[7]
5. ENDIF

RETURN result

## Flowchart: LE8\_11 Days of the Week



# LE8\_12 Search Element

Write a function to search an element in array using pointers and return the index location. The function accepts the starting address of the array, number of entries and number to search.

Enter number of entries: 5

Enter number: 9

Enter number: 7

Enter number: 5

Enter number: 3

Enter number: 1

Search data: 5

FOUND in Index 2

Search data: 8

NOT FOUND

## Pseudocode: LE8\_12 Search Element

main()

START

1. INITIALIZE size and search as integer
2. PROMPT and GET size
3. INITIALIZE list[size] as integer
4. FOR i = 0 to size exclusive by 1
   1. PROMPT and GET list[i]
5. ENDFOR
6. PROMPT and GET search
7. CALL module, result = find(list, size, search)
8. IF result is not equal to -1
   1. PRINT result
9. ELSE
   1. PRINT “NOT FOUND!”
10. ENDIF

STOP

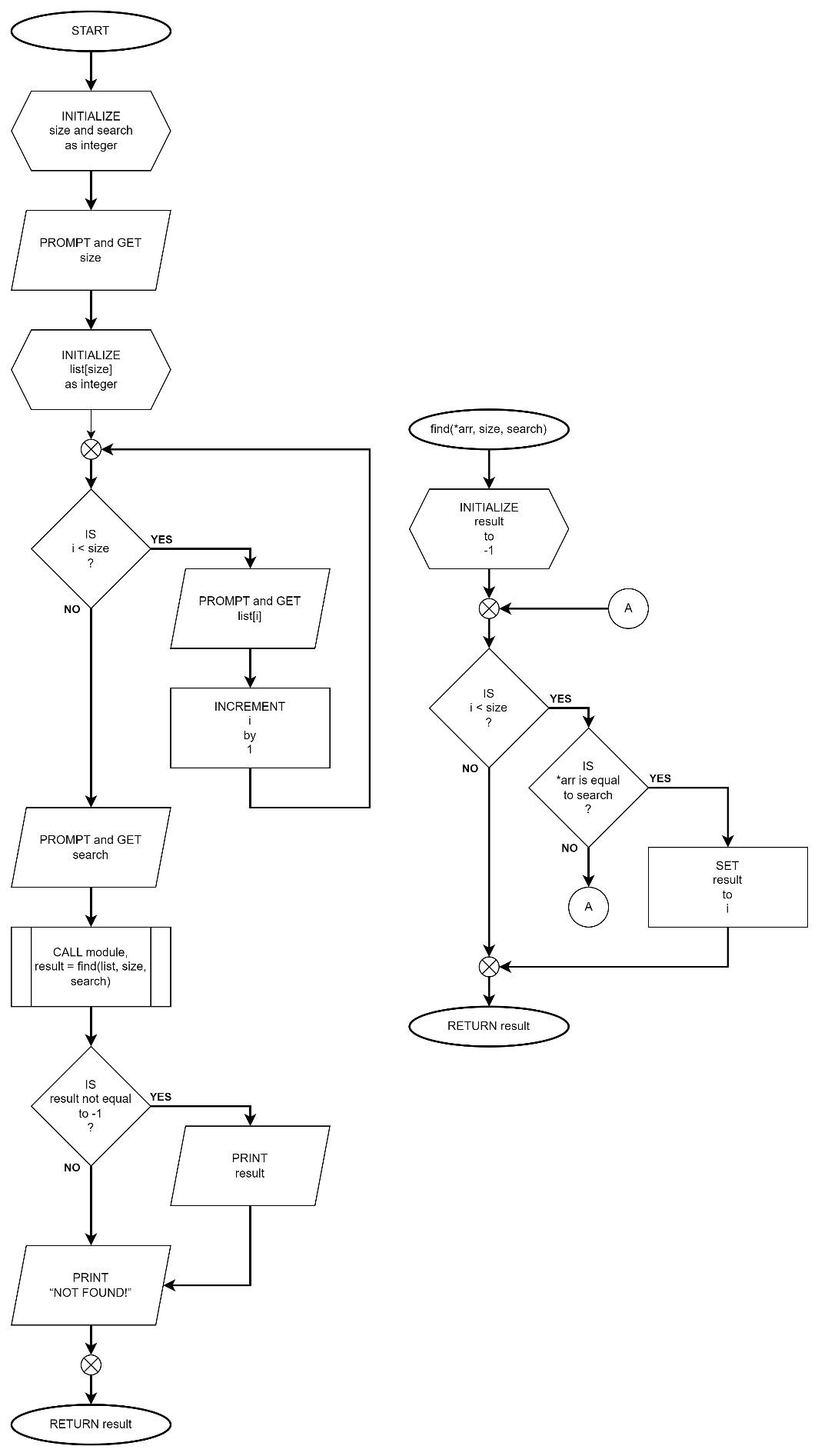
find(\*arr, size, search)

START

1. INITIALIZE result to -1
2. FOR i = 0 to size exclusive by 1
   1. IF \*arr is equal to search
      1. SET result to i
      2. BREAK from LOOP
   2. ENDIF
   3. INCREMENT arr by 1
3. ENDFOR

RETURN result

## Flowchart: LE8\_12 Search Element



# LE8\_13 Maximum Number

Create a function that determine and returns the maximum number in an array of integers using pointers. The function accepts the starting address of the array and number of entries.

Example output:

Enter number of entries: 5

Enter number: 3

Enter number: 7

Enter number: 9

Enter number: 1

Enter number: 5

Maximum Number: 9

## Pseudocode: LE8\_13 Maximum Number

main()

START

1. INITIALIZE size as integer
2. PROMPT and GET size
3. INITIALIZE list[size] as integer
4. FOR i = 0 to size exclusive by 1
   1. PROMPT and GET list[i]
5. ENDFOR
6. CALL module, result = largestNumber(list, size)
7. PRINT result

STOP

largestNumber(\*arr, size)

START

1. INITIALIZE large as \*arr
2. INCREMENT arr by 1
3. FOR i = 1 to size exclusive by 1
   1. IF \*arr is greater than large
      1. SET large to \*arr
   2. ENDIF
   3. INCREMENT arr by 1
4. ENDFOR

RETURN large

## Flowchart: LE8\_13 Maximum Number

