<Advanced C Programming and Lab> Ch 9. Pointers

※ Note

- If not mentioned, assume that there is no additional inputs.
- If not mentioned, do not print a space in the beginning and end of each line.
- In input and output examples, after ↦ symbol is to explain the input and output.
- In output examples, □ symbol indicates a space.
- ▶ indicates the requirements for pointers.

Section 1 [Problem 1] Declare an int type variable x and an int type pointer variable px.

- Read a value of x using scanf().
- Change the value of x using px and scanf().
- Print the modified value using px.

Input Example 1	Output Example 1

`5      ↦ x`
`10     ↦ change to this`
	`10       ↦ Modified value`

Section 1 [Problem 2] Read two positive integers N and M. Print the sum of the integers from N to M.

- N < M
- Declare pointer variables for all the variables and store their addresses to the pointer variables.
- Use pointer variables to compute the sum.

▶ All variables should be pointer variables except arrays

▶ Declare a normal variable and a pointer variable.

▶ Assign the address of the normal variable to the pointer variable.

▶ Use the pointer variable to access and use the normal variable.

Input Example 1	Output Example 1
`1 100`	`5050`

Input Example 1	Output Example 1
`3 7`	`25`

Section 2 [Problem 3] Declare an int type variable arr of size 100. Read N integers (N<=100). Read an index from a user. Using a pointer variable, print the value of the element that is corresponding to the received index.

▶ All variables should be pointer variables except arrays

▶ Cannot use square brackets to use arrays.

Input Example 1	Output Example 1
5 ↦ N 2 4 7 9 13 3 ↦ M	9

Section 2 [Problem 4] Declare an int type variable arr of size 100. Read N integers (N<=100). Using a pointer variable(s), print the minimum and maximum value of the array.

▶ All variables should be pointer variables except arrays

▶ Cannot use square brackets to use arrays.

Input Example 1	Output Example 1
5 ↦ N 10 4 7 27 1	1 27

Section 2 [Problem 5] Read a number of characters (>=1 and <=20) until receiving '#'. Print the characters in a reverse order using a pointer variable(s).

▶ All variables should be pointer variables except arrays

▶ Cannot use square brackets to use arrays.

Input Example 1	Output Example 1
duck#	kcud

Input Example 2	Output Example 2
duck#pond	kcud

Section 2 [Problem 6] Read 10 alphabet letters. Print the most frequent character and its frequency using a pointer variable(s). If two or more such characters exist, print the character that appears earlier.

▶ All variables should be pointer variables except arrays

▶ Cannot use square brackets to use arrays.

▶ In a loop, use the address of a variable.

Input Example 1	Output Example 1
domination	i 2

Section 2 [Problem 7] Declare an int type array of size 3. Initialize it using the values that receive from a user. Print the median of the 3 values.

- Use an array int x[3].
- Do not use x[0], x[1], x[2]. Use a pointer variable p.
- Do not use additional variables.
- median is the value that appear in the middle when ordered the values in ascending order.
- Example: 1 9 7, order in ascending order: 1 7 9, median is 7.

▶ All variables should be pointer variables except arrays

▶ Cannot use square brackets to use arrays.

▶ In a loop, use the address of a variable.

Input Example 1	Output Example 1
1 9 7	7

Input Example 2	Output Example 2
-7 10 0	0

Section 2 [Problem 8] Read 5 integers from a user and store in an array arr[]. Store and print the rank of the integers as ordered in descending order.

- Do not the expression arr[i] and rank[i]. Use two pointer variables p and q that point to the arrays.
- Example: 1 6 4 9 1, desceding order: 9 6 4 1 1, there are two intergers that are ranked 4th.

▶ All variables should be pointer variables except arrays

▶ Cannot use square brackets to use arrays.

▶ In a loop, use the address of a variable.

Input Example 1	Output Example 1
1 2 3 4 5	1=r5 2=r4 3=r3 4=r2 5=r1

Input Example 2	Output Example 2
1 6 4 9 1	1=r4 6=r2 4=r3 9=r1 1=r4

Section 2 [Problem 9] Read an integer N which is the size of an array. Read N integers. Count and print the number of integers that are received before 0 was received.

- N<=50
- 0 only appears once among N integers.
- Integers (except 0) can appear multiple times.
- Declare and use pointer variables for the normal variables that are declared and used.
- Use pointer variables to compute the number integers.

▶ All variables should be pointer variables except arrays

▶ Cannot use square brackets to use arrays.

▶ In a loop, use the address of a variable.

Input Example 1	Output Example 1
10 ↦ N 4 5 8 9 8 1 0 1 9 3	6

Input Example 2	Output Example 2
5 0 1 3 -2 -4	0

Section 4 [Problem 10] Read N integers (N ≤ 50) and store them in an array. Read two indices a and b. Then, swap the two elements that are accessed by the indices a and b. Print the elements of an array.

- a < b or a > b
- Define a function swap( )
  - ○ Parameter: two int pointers
  - ○ Return: None
  - ○ Swap two values
- main( ) receives inputs and prints outputs.

Input Example 1	Output Example 1
6 ↦ N 3 2 0 1 4 6 2 4 ↦ a b	□3 2 4 1 0 6

Section 4 [Problem 11] Read two positive integers and print their sum.

- Defin a function sum( )
  - ○ Parameter: one int pointer, two integers

  ○ Store the sum of two integers where the pointer variable points out

  ○ Return: none

- main( ) receives inputs and prints outputs.

Input Example 1	Output Example 1
30 20	50

Input Example 2	Output Example 2
17 7	24

Section 4 [Problem 12] Read N integers (N ≤ 100) and print the number of even numbers.

- Define a function findeven( )
  - ○ Parameter: int array, size of the array
  - ○ Return: the number of even numbers
- main( ) receives inputs and prints outputs.

Input Example 1	Output Example 1
5 ↦ N 10 3 8 13 20	3

Section 4 [Problem 13] Read 6 characters and store them in an array. Copy the characters to another array.

- Define a function strcopy( )
  - ○ Parameter: two character arrays a[ ] and b[ ]
  - ○ Copy the characters in an array b[ ] to another array a[ ]
  - ○ Return: none
- main( ) receives inputs and prints outputs.

Input Example 1	Output Example 1
beyond	beyond

Section 4 [Problem 14] Read N integers (N ≤ 100) and calculate and print the total sum.

- Define a function arrsum( )
  - ○ Parameter: int array, size of the array
  - ○ Return: total sum
- main( ) receives inputs and prints outputs.

Input Example 1	Output Example 1
5 ↦ N 3 10 15 20 27	75

[Problem 15] Read 10 integers, sort them in descending order, print the sorted integers.

- main( ) function
  - ○ Store the inputs in an int array
  - ○ Call ABC() 9 times using the int array and index k (k=0,1,2,...)
  - ○ Print the integers in an array

- Define a function ABC( )
  - ○ Parameter: int array, integer k
  - ○ Find the largest element between kth element and the last element in the array. Swap the largest element and kth element.
  - ○ Return: none

Input Example 1	Output Example 1
1 3 5 7 9 2 4 6 8 10	□10 9 8 7 6 5 4 3 2 1

Input Example 2	Output Example 2
13 56 27 89 43 76 32 68 91 8	□91 89 76 68 56 43 32 27 13 8

[Problem 16] Read N integers (N ≤ 20). Print the integers in ascending or descending order.

▶ All variables should be pointer variables except arrays

▶ Cannot use square brackets to use arrays.

▶ In a loop, use the address of a variable.

- Define a function userAlign( )
  - ○ Parameter: an int pointer variable for an array, size of the array, an integer that indicates the type of sorting (0: ascending, 1: descending)
  - ○ Sort the integers in ascending or descending order
  - ○ Return: none
- main( ) receives inputs and prints outputs.
- No repeated inputs

Input Example 1	Output Example 1
5 0 ↦ 5 integers, ascending order 5 7 9 4 3 ↦ N positive integers	□3 4 5 7 9

Input Example 2	Output Example2
3 1 ↦ 3 integers, descending order 3 8 9	□9 8 3

[Problem 17] Read N integers (N ≤ 20) twice. Add them in reverse order (see below).

- ▶ All variables should be pointer variables except arrays
- ▶ Cannot use square brackets to use arrays.
- ▶ In a loop, use the address of a variable.

- Define a function addArray( )
  - ○ Parameter: three int pointers for three arrays, size of the arrays
  - ○ Add the first element in the first array to the last element in the second array, add the second element in the first array to the second element from the last element in the second array, and so on. Store the results in the third array
  - ○ Return: none
- main( ) receives inputs and prints outputs.

Input Example 1	Output Example 1
3 ↦ size of an array 1 2 3 5 10 15	□16 12 8 ↦ 16=1+15, 12=2+10, 8=3+5

Input Example 2	Output Example 2
4 ↦ size of an array 3 8 9 5 0 1 -5 6	□9 3 10 5