Computer Programming and Applications
EC1011301, Fall 2022

---

*When you get your flash drive, please **manually** create a file named as your ID and Chinese name, such as "B11132099 黃意婷" to make sure your answer responses stored in THIS flash drive. This is your responsibility to ensure your answer responses delivered correctly and successfully. Also, absolutely no speaking is allowed during tests. If you have a question, please raise your hand. Any evidence of dishonest work will have an impact on the overall grade. Good luck!*

There are a set of C++ source files in your flash drive for each corresponding test item. Please directly edit it as your answer responses.

---

1. (5%) Write a program (myID.cpp) to create a text file (ID.txt) which contains your student ID, a space, and your English name.

   An example:

   B11132099.txt

   B11132099 Yi-Ting Huang

2. (5%) Write a function **templated** called maximum in myMax.cpp to determine the larger of two arguments and return the largest one. In myMax.cpp, you can test the program using integer, character and floating-point number arguments.

3. (10%) Write a program (myMajor.cpp) to describe ten CEECS students choosing their majors by random, and print out their major. Please follow the instruction.
   First, use enum class to list three majors, that is, "Electronic and Computer Engineering", "Electrical Engineering" and "Computer Science and Information Engineering" (readable abbreviations are acceptable).
   Second, design a function getMajor to randomly select one of the three majors.
   Third, write a function including a switch statement to print out "Student #: #mood# Welcome to #major#." based on the following table.
   Finally, let each student choose his/her major via function getMajor and display his/her result on the screen.

Mood	Major
Wonderful!	Electronic and Computer Engineering
Excellent!	Electrical Engineering
Amazing!	Computer Science and Information Engineering

An example

```
Student 1: Excellent! Welcome to Electrical Engineering.
Student 2: Amazing! Welcome to Computer Science and Information Engineering.
Student 3: Wonderful! Welcome to Electronic and Computer Engineering.
Student 4: Excellent! Welcome to Electrical Engineering.
Student 5: Wonderful! Welcome to Electronic and Computer Engineering.
Student 6: Excellent! Welcome to Electrical Engineering.
Student 7: Wonderful! Welcome to Electronic and Computer Engineering.
Student 8: Amazing! Welcome to Computer Science and Information Engineering.
Student 9: Excellent! Welcome to Electrical Engineering.
Student 10: Amazing! Welcome to Computer Science and Information Engineering.
```

4. (5%) In mySwap1.cpp, edit the program to use **pass by reference function** to swap two integers.

5. (5%) In mySwap2.cpp, edit the program to use **pass by reference function with pointers** to swap two integers.

6. (10%) Write a program (myReverseArray.cpp) to reverse a given **built-in** integer **array** with six elements using the pointers. First, declare an integer pointer to the given built-in integer array {1, 2, 3, 4, 5, 6}. Next, respectively print out the original elements and the reversed elements of the array only using the pointers.

   Output:

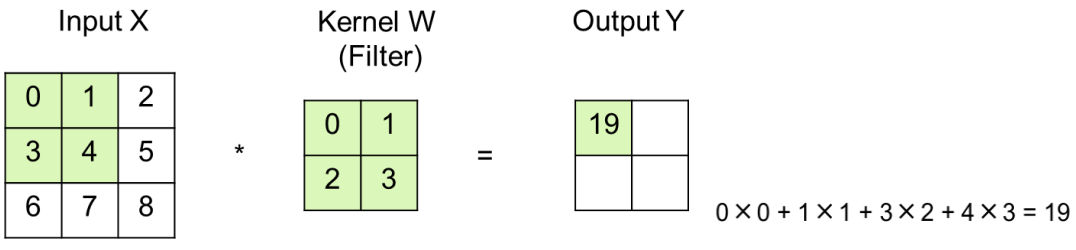
```
Original array: 1, 2, 3, 4, 5, 6
Reversed array: 6, 5, 4, 3, 2, 1
```

7. (15%) Write a program in myCNN.cpp to implement two-dimensional cross-correlation operation. Considering the following instruction, create one two-dimensional array with a height of n_h and width of n_h as an input matrix X and the other one two-dimensional array with a height of k_h and width of k_w as a kernel matrix W. For example, you can see a 3×3 input matrix X and a 2×2 kernel matrix W in Step 1.
   In the two-dimensional cross-correlation operation, we begin with the convolution window positioned at the upper-left corner of the input matrix X and slide it across the input matrix X, both from left to right and top to bottom. Here, we defaulted to sliding one element at a time. When the convolution window slides to a certain position, the input sub-matrix (as the green region of the input matrix X) contained in that window and the kernel matrix W are multiplied elementwise and the result is summed up yielding a single scalar value. This result gives the value of the output matrix Y at the corresponding location.
   In our instruction, the output matrix Y has a height of 2 and width of 2 and the four elements are derived from the two-dimensional cross-correlation operation.
   For example, in Step 1, the first element 19 of output matrix Y is derived from $0 \times 0 + 1 \times 1 + 3 \times 2 + 4 \times 3 = 19$ where (0, 1, 3, 4) from X and (0, 1, 2, 3) from W.
   Next, the convolution window slides to the next position until the convolution window (the green region) slides the input matrix X over all locations and gives all values of the output matrix Y at the corresponding locations (as Step 2, 3 and 4).
   Finally, please print out the output matrix Y on the screen.

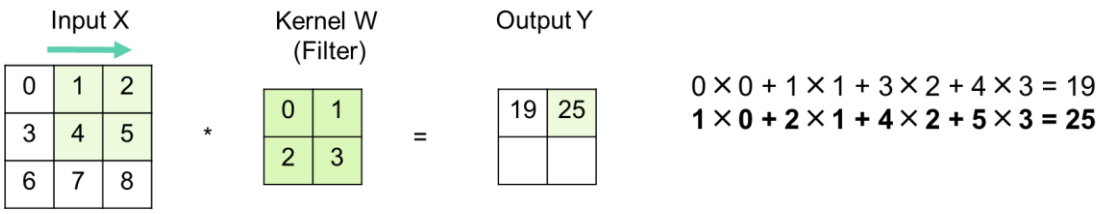
   Please note that the input matrix X ($n_h \times n_w$), kernel matrix W ($k_h \times k_w$), and output matrix Y ($n_h - k_h + 1$, $n_w - k_w + 1$) are square.
   You can implement array with `int a[]` or `#include<array>` and initialize these matrices using values in the following test case.

   Instruction:

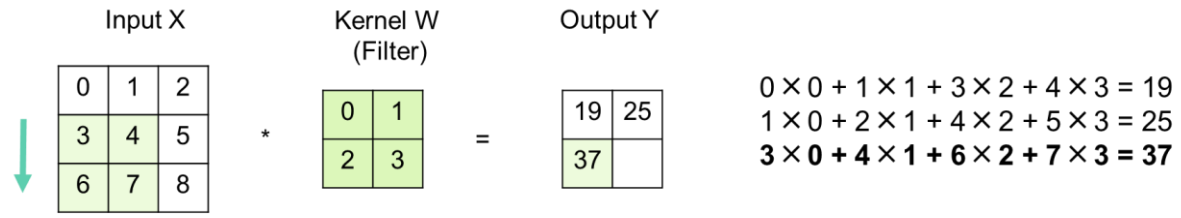
Step 1:



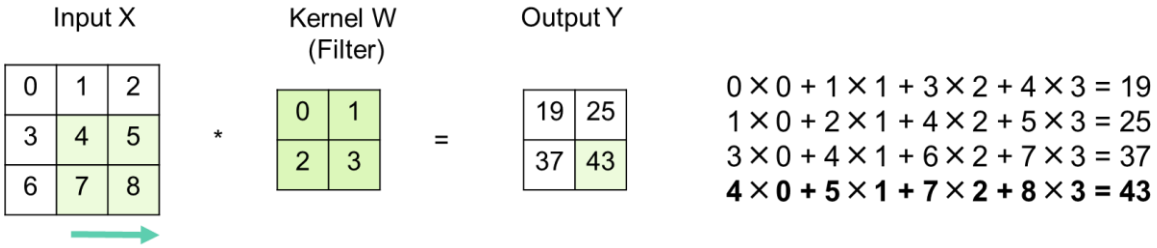
Step 2:



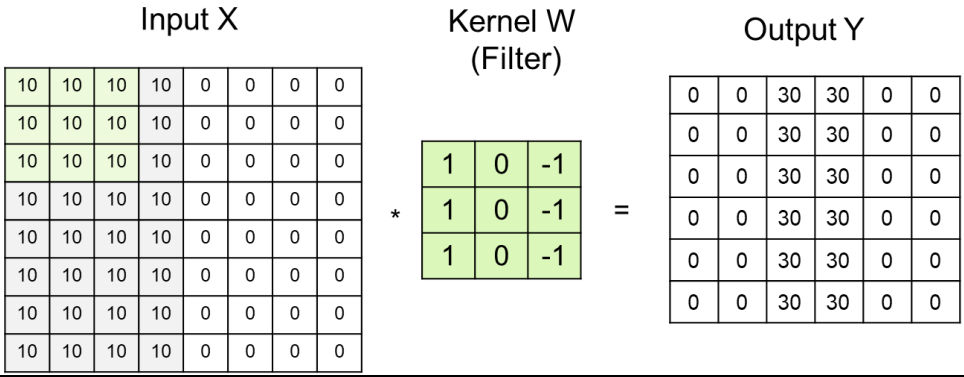
Step 3:



Step 4:

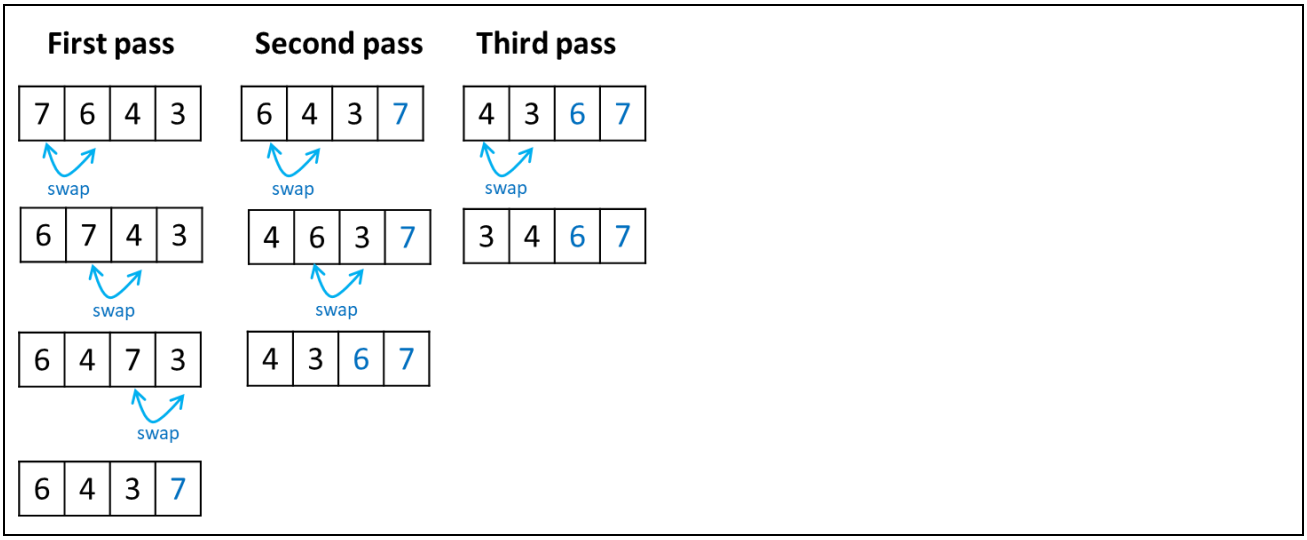


Output:



8. (15%) Write functions to implement "Bubble sort" in mySort.cpp for sorting a given unsorted array of positive integers in **ascending order**.
Bubble sort is a sorting algorithm that compares two adjacent elements and swaps them until they are in the intended order. That is, Step 1 starts with the first element and Step 2 compares the current element with the next element. Step 3 make a comparison between these two elements. If the current element is greater than the next element, then swap both the elements. If not, move to the next element. Repeat steps 1 – 3 until we get the sorted list.
Take an example in the following instruction, the list that contains the elements 7, 6, 4, 3 initially. In bubble sort, to sort a list of size *n*, we need to perform *n* – *1* iterations. Notice we had 4 elements in this example, so we got the sorted list in 3 passes. In the first pass, we make four comparisons and three swaps if necessary. In the second pass, we make three comparisons and two swaps if necessary. In the third pass, we make two comparisons and one swap.

Instruction:



First, write a function named function1 to use C++ Standard Library #include <algorithm> to sort the elements.
Second, write a function named function2 to sort the elements **without** loading C++ Standard Library.
Third, write a **recursive** function named function3 to sort the elements.

You can implement array with int a[] or #include<array>. Please note that you can comment function1, function2 and function3 in main() when developing your code.

An example:

```
unsortedArray = {4, 5, 1, 2, 3, 6, 11, 15, 13, 14, 8, 9, 7, 10, 12}
sortedArray = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}
```

9. (20%) Write a program (myRandomMask.cpp) to process the ten files (1.txt, 2.text, …, 10.txt) in your flash drive. For each file, there are a set of positive numbers from 1 to 10. Please write a program to **randomly select 10% of numbers, replace** them with the value **"0"** and **create additional files** for the replacement with the name "data_#.txt", such as "data_1.txt".
   You can follow the following instruction.
   First, read file contents into a vector or an array.
   Second, decide 10% of contents (10% of integers) for the replacement and then replace the numbers with "0".
   Third, create a new file for write the modified contents.

   A toy example:

Input	Output
1.txt 1 2 3 4 5 6 7 8 9 10	data_1.txt 1 2 0 4 5 6 7 8 9 10

10. (10%) Write an exception-handling statement for throwing an out_of_range expecption in myOutOfRange.cpp to notify an attempt to use out-of-range subscript in a vector. Also, please print out the error message on console.

It is the end of test sheet. Before you hand in your response, please check your output files carefully.

0	*#name#.txt*
1	*myID.cpp* *#ID#.txt*
2	*myMax.cpp*
3	*myMajor.cpp*
4	*mySwap1.cpp*
5	*mySwap2.cpp*
6	*myReverseArray.cpp*
7	*myCNN.cpp*
8	*mySort.cpp*
9	*myRandomMask.cpp* *data_1.txt data_2.txt data_3.txt data_4.txt data_5.txt data_6.txt data_7.txt data_8.txt data_9.txt data_10.txt*
10	myOutOfRange.cpp