Computer Programming 1

Laboratory 14

Topics of the exercises

- Strings
- Command-line arguments
- Files management (read and write from/to file)
- Matrix

Strings

```
char stringa [] = “Orange”;
```

- It contains 7 elements, the characters of the string and the terminator ‘\0’ that is added automatically;
- Operators of input e output operates directly on the strings

```
char buffer[256];
cin >> buffer;
cout << buffer;
```

String: the library *<cstring>*

Functions of the library <cstring>

- strlen(s): it returns the length of the string s;
- strcpy(d, s): copies the string s inside the string d and returns d;
- strncpy(d, s, n): copies the first n characters of string s inside the string d and returns d;
- strcmp(d, s): returns a negative, NULL, or a positive value if *s* is alphabetically less, equal to, or greater than d;
- strncmp(d, s, n): returns a negative, NULL, or a positive value if the first n characters of *s* are alphabetically less, equal to, or greater than d;
- strcat(d,s): appends a copy of s to d and returns d;
- strncat(d,s,n): appends a copy of the first n character of s, plus a terminating null characrer, to d and returns d;

The command line (1)

Command-line arguments:

```
int main( int argc, char * argv [] )
```

argc indicates the total number of command-line words (arguments + the name of the executable)

argv is a multidimensional array that contains arguments in the form of strings;

The command line (2)

E.g.,

./a.out input.txt output.txt 56

argc = 4

argv: [“./a.out”, “input.txt”, “output.txt”, “56”]

To eventually convert strings in numbers, we need to use the functions: atoi() o atof() of the library <cstdlib>

I/O Stream and text Files (1)

The functions to operate on files are available through the library <fstream>

- Using a stream is analogous to cin and cout (so using the operators >> and <<)

Note: remember to close a stream once opened

I/O Stream and text Files (2)

- Open and close a stream

```
1. fstream input, output;
2. input.open("file.txt", ios::in);
3. output.open("output.txt", ios::out);

4. input.close();
5. output.close();
```

I/O Stream and text Files (3)

- Read and write a file

```
1. char buffer[256];
2. input >> buffer;
3. while(!input.eof())
4. {
5.   input >> buffer;
6. }
7. output << buffer;
```

I/O Stream and text Files (4)

Functions of the library <iostream>

input.get(c): reads a character (including spaces) from the stream and stores it in variable c;

input.eof(): indicates if end of the stream is reached;

input.fail(): indicates if there were any errors in opening the stream;

- …

Exercises and their execution

In the rest of the slides, you can find 9 exercises.

(1) Try to do them without using additional material (access to internet, slides, other code, etc.)

(2) Try to do them by allocating the indicated maximum amount of time:

- 1h max for: Exercise 1 and Exercise 2
- 30 min max for: Exercise 3 and its variants (3b and 3c)
- 1h max for: Exercise 4, Exercise 5 (5b) and Exercise 6
- 1h max for: Exercise 7 (5b) and Exercise 8
- 30 min max for: Exercise 9 (9b)

Exercise 1: String

- Write a program counts the number of worlds in a string (brief sentence) given in input from the user
- Add checks on the input string/sentence to avoid errors
- You can assume a max length for the input string

E.g.,

- Input string: This is a brief sentence
- Expected output: Number of words: 5

Exercise 2: String Permutations

- Write a program that, given a string as input from the user, prints all permutations of a string with duplicates and the number of such permutations
  - We recall that, given n characters, the number of permutations is n!
  - You can print the expected number of permutations (i.e., n!) and the number of actually computed permutation (these two number are expected to be equal)
  - You can fix a maximum length of the input string (e.g., 4 or 5)
  - You can check that the inserted string contains only valid characters

- E.g., given the string ABCD

  n=4 → nr. Permutations: 4!=24

ABCD	BCAD	CDAB
ABDC	BCDA	CDBA
ACBD	BDCA	DBCA
ACDB	BDAC	DBAC
ADCB	CBAD	DCBA
ADBC	CBDA	DCAB
BACD	CABD	DACB
BADC	CADB	DABC

Example 3: Learning how to copy

- Write a program that, given as input via command-line, the names of two files, A and B, copy the contents of A within B.
- Implement also some controls that warn the user if:
  - the number of arguments is wrong (the input file is missing)
  - if the input file does not exist

Documentation for <fstream>: http://www.cplusplus.com/reference/fstream/fstream/

Example 3b: Learning how to copy

- Write a program that, given as input via command-line, the names of two files, A and B, copy the contents of A within B.
- Implement also some controls that warn the user if:
  - the number of arguments is wrong (the input file is missing)
  - if the input file does not exist
- Constraint: do not use function like gets(..), fgets(..), get(ch), getline(..), but use only stdin and stdout streams

Documentation for <fstream>: http://www.cplusplus.com/reference/fstream/fstream/

Example 3c: Learning how to copy

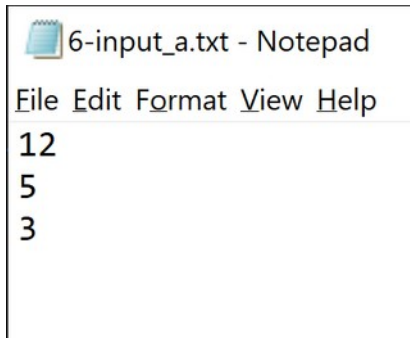
- Write a program that, given as input via command-line, the names of three files, A, B and C, copy the contents of A and B into C.
- Implement also some controls that warn the user if:
  - the number of arguments is wrong (the input file is missing)
  - if the input file does not exist
  - If the file C already exist, in such a case, ask to the user if (1) the C file needs to be overwritten, if (2) it needs to be preserved and thus the new content needs to be added at the end of the existing C file content, or if (3) the copy operation has to be cancelled

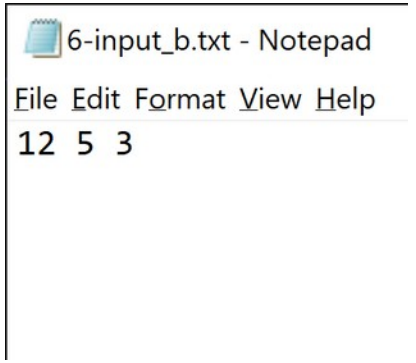
Documentation for <fstream>: http://www.cplusplus.com/reference/fstream/fstream/

Example 4: Sum integers in file

- Write a program to calculate the average of numbers stored in a file.

- Constraints:
  - The name of the file has to be passed as parameter from the command line
  - The input file can be in two formats like the following images





Example 5: letter substitution

- Write a program that, given in input the name of a file F and a letter L, prints the contents of that file on the screen, replacing every occurrence of the letter L with the symbol "?“

E.g.,

"What a magnificent day", a

→

"What m?gnific? day?t?"

Example 5b: letter substitution

- Write a program that, given in input the name of a file F and three letters L1, L2, L3, prints the contents of that file on the screen, replacing every occurrence of the letters L1, L2, L3 respectively with the symbols: “?”, “!”, “#”

E.g.,

"What a magnificent day", a

→

"What m?gnific? day?t?"

Example 6: Intersection of words

- Write a program that, given as input two files A and B from the command line, generates a third C file that contains all the words present in both files A and B.

- It is allowed to:
  (i) use strcmp() and strcpy() functions of the library <cstring>
  (ii) open a maximum of one stream per file
- Assumptions:
  (i) the files contain a maximum of 1000 words
  (ii) the maximum length of the individual words is 100

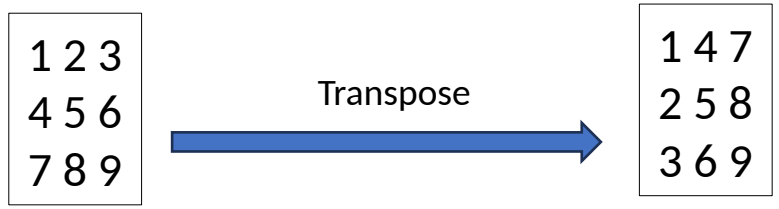
Example 7: Matrix transpose

- Write a program that finds the Transpose of a matrix in which:
  1. the maximum number of rows and columns is given by the user in the range [1,10]
  2. the values of the coefficients of the matrix cells are (integers) given by the user

Constraints:

- Use 2-dim array
- Add checks for the ranges

Rows=3
Cols=3
Values= 1 2 3 4 5 6 7 8 9

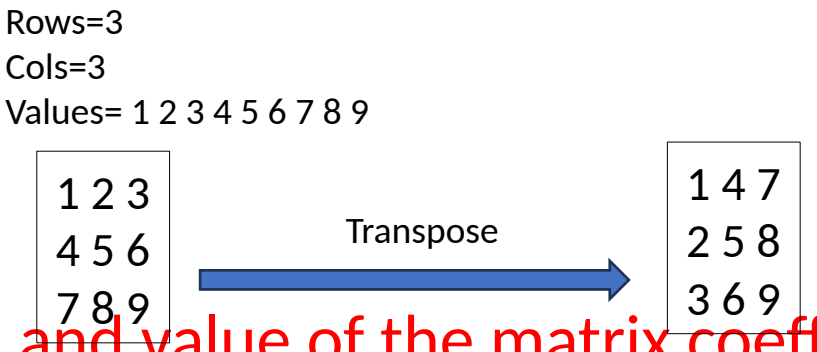


Example 7b: Matrix transpose

- Write a program that finds the Transpose of a matrix in which:
  1. the maximum number of rows and columns is given by the user in the range [1,10]
  2. the values of the coefficients of the matrix cells are (integers) given by the user

Constraints:

- Use 2-dim array
- Add checks for the ranges



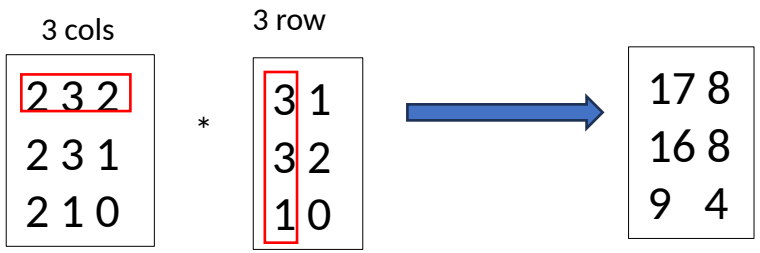
Variant: Read the number of rows, cols, and value of the matrix coefficients from a file

Example 8: Matrix multiplication

- Write a program that multiplies two matrices, and in which:
  1. the maximum number of rows and columns of the two matrices are given by the user in the range [1,10]
  2. the number of columns of the first matrix needs to be equal to the number of rows of the second matrix
  3. the values of the coefficients of the matrix cells for both matrices are (integers) given by the user

Constraints:

- Use 2-dim array
- Add checks for points (1) and point (2)



2*3+3*3+2*1=17
2*1+3*2+2*0=8

2*3+3*3+1*1=16
2*1+3*2+1*0=8

2*3+1*3+0*1=9
2*1+1*2+0*0=4

Example 9: Armostrong number

- Write a program that checks if a number is an Armstrong number or not
  - An Armstrong number is an n-digit number that is equal to the sum of the nth powers of its digits.
  - E.g., $407 = 4^3 + 0^3 + 7^3 = 64 + 0 + 343 = 407$ → YES
  - E.g., $29 = 2^2 + 9^2 = 4 + 81 = 85$ → NO

Constraints:

- The input number to check is given by the user
- Add a check to verify that the input number is actually an integer number

Example 9b: Armostrong number

- Write a program that checks if a number is an Armstrong number or not
  - An Armstrong number is an n-digit number that is equal to the sum of the nth powers of its digits.
  - E.g., $407 = 4^3 + 0^3 + 7^3 = 64 + 0 + 343 = 407$ → YES
  - E.g., $29 = 2^2 + 9^2 = 4 + 81 = 85$ → NO

Variant:

- Implement a function that checks a number and returns true if is an Armstrong number
- Read 10 numbers from a file, and store them in an array
- Check 10 numbers of the array