REPORT

Assignment 2 (Part 1 of the project)

Aubakirov Talgat

Omar Adilkhan

BDA1902

“The window counter considered numbers often. How many tons of air did the universe contain, for instance. She wanted to know the average number of thoughts projected on an object in its life, to measure the silences in a dream, to calculate the ideal amount of light a window should emit. But the most pressing question of all was the number of windows the world contained. How many worlds could a person view from within or without?”

― Meia Geddes, [The Little Queen](https://www.goodreads.com/work/quotes/57028941)

Goal: C++/Python/Java (or another language) program finds all symbols (not only letters) in this text and computes their probabilities out of all symbols. The most frequent symbols will have higher probabilities.

Introduction: The majority of writers are suffering about counting the letters and symbols in their text. The main task of this project to prevent confusion of writers and get probability of each symbol in written article, essay and other related text documents. All work will be done in C++ programming language.

1. Main theory
   1. ASCII-7:

ASCII – abbreviated from American Standard Code for Information Interchange, is a character encoding standard for electronic communication. ASCII codes represent text in computers, telecommunications equipment, and other devices. Most modern character-encoding schemes are based on ASCII, although they support many additional characters. In 1.1. table can be seen that every registered symbol have own assigned code.

In C++ it commonly used ASCII-7 coding system. It is much more easier to assign every single element of array for each symbol. The max number of symbol in ASCII-7 is 128. For example, 65th element for capital a, 66th for capital b and it continues in such logic.

int\* arr = new int[128];

The code above is to create dynamic array for each possible symbol in ASCII-7. It means that if programmer looking for number of occurrence of capital a, he/she should write 65 in index of array.

std::cout << arr[65]; // For capital a

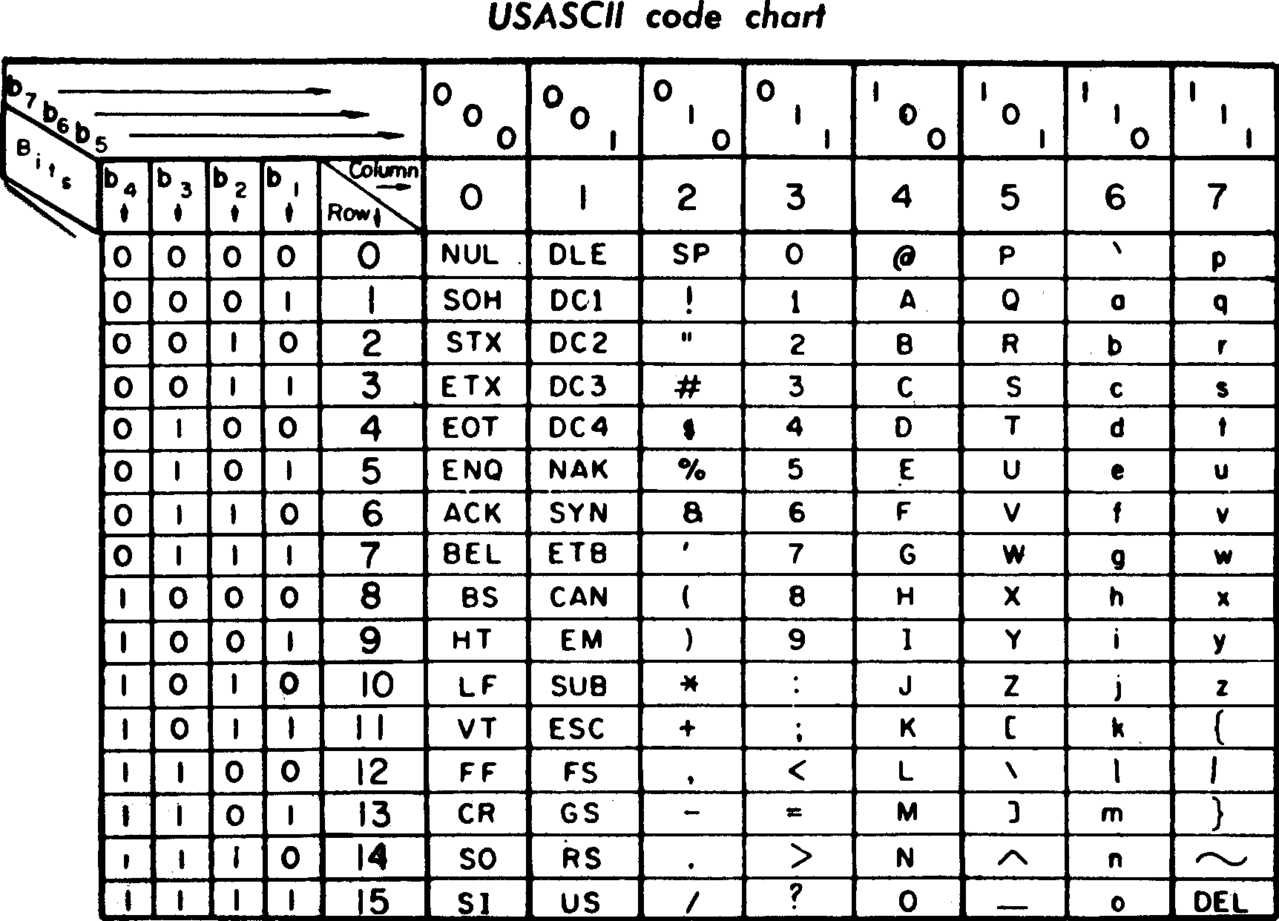


Table 1.1.1. ASCII table

* 1. Insertion sort:

Insertion sort is a simple sorting algorithm that works similar to the way you sort playing cards in your hands. The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in the sorted part. The visualization of insertion sort example is given in picture 1.2.1.



Picture 1.2.1. Insertion sort example

1. Implementation
   1. Prototyping main functions:

There are main functions that will be used in project: Sorter, counter and file opener.

int\* uniqueSorter(int\*);

int\* counter(string);

string file\_read(string);

The code above is prototypes of the function that will be implemented below.

* 1. Libraries used:

Majority of functions, methods that used in project are already implemented in libraries. There only 2 libraries are used in current project: iostream and fstream.

* iostream for input and output in console
* fstream for input and output in text file

The code below for importing libraries listed above

#include <iostream>

#include <fstream>

* 1. Implementing counter function:

It is already known that every symbol has own code. In this project it is sufficient just incrementing elements in array, that indexes current symbol in text. It is enough to apply this method for every symbol in text.

Code below for implementation:

int\* counter(string text) {

int\* arr = new int[128];

for(int i = 0; i < 128; i++) {

arr[i] = 0;

}

for(int i = 0; i < text.length(); i++) {

arr[text[i]]++;

}

return arr;

}

* 1. Implementing sorter for array of symbols:

From implementation of counter, it can be got the array of number of each symbol in text. Insertion sort can be applied for sort values in got array, however the order will be changed and it never be known what symbol was assigned for each number. For that reason, it should be created another array for saving the order of symbols, but in this project that data stored in same array as resulting from previous function.

int\* uniqueSorter(int\* counted) {

int\* arr = new int[256];

for(int i = 0; i < 128; i++) {

arr[i + 128] = i;

arr[i] = counted[i];

}

for(int i = 0; i < 128; i++) {

for(int j = 0; j < i; j++) {

if(arr[i] > arr[j]) {

swap(arr[i], arr[j]);

swap(arr[i + 128], arr[j + 128]);

}

}

}

return arr;

}

Code above for implementation of current function.

* 1. Implementation of file reader function:

For this function, the fstream was used. First of all, the program will open file. Then, read it line by line and save into main string.

string file\_read(string name) {

string text, line;

ifstream myfile (name);

if (myfile.is\_open())

{

while(getline(myfile,line)) {

text += line + '\n';

}

myfile.close();

}

return text;

}

Code above for implementation of current function.

* 1. Implementation of main:

In main function, all other function above was used. There are special way of using functions and outputting data from resulting array:

int main() {

string text = file\_read("text.txt");

int\* arr = uniqueSorter(counter(text));

double size = text.length();

for(int i = 0; i < 128; i++) {

if(arr[i] == 0) {

break;

}

cout << char(arr[i + 128]) << " - " << arr[i] << endl;

}

}

1. Testing
   1. Preparation text for test:

The text was taken from random text generator website: <https://www.blindtextgenerator.com/lorem-ipsum>

The text is stores in “text.txt” file

* 1. Running the code:

The result of run:

- 0.146324

e - 0.0947

i - 0.0793899

u - 0.0764187

s - 0.0694076

t - 0.0645775

a - 0.0628211

l - 0.0502042

n - 0.0480526

r - 0.0443495

m - 0.0360065

o - 0.0339573

c - 0.0323327

. - 0.0228041

d - 0.0209306

p - 0.0195987

, - 0.0193645

g - 0.011607

v - 0.0112411

b - 0.0107727

q - 0.00977737

f - 0.00522533

h - 0.00490332

P - 0.00440567

N - 0.00281026

C - 0.00225407

S - 0.00222479

V - 0.00212233

D - 0.00185887

A - 0.00143441

M - 0.00137586

F - 0.00134658

I - 0.00128804

E - 0.000980665

j - 0.000878207

Q - 0.000629382

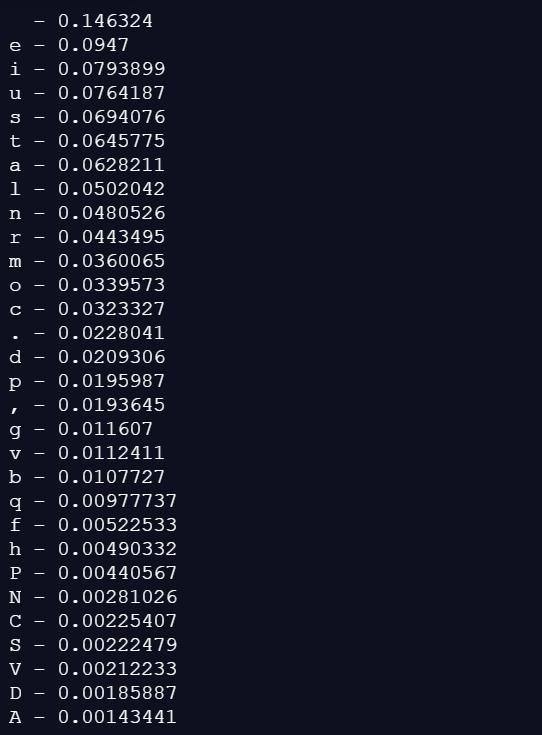
U - 0.000614745

y - 0.000556198

; - 0.000322009

L - 0.000117094

/n - 1.46368e-05



1. Conclusion:

This program possibly can make work of majority people more effective and uncomplicated. It was exemplary experience to do such project.