НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

із лабораторної роботи №*3*

із дисципліни «Основи програмування»

на тему

«Опрацювання символьних даних»

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-73 | *асистент Громова В.В.* |
| *Садченко М.В.* |  |

Київ — 2018

ЗМІСТ

[ВСТУП 3](#_Toc508677009)

[1 ПОСТАНОВКА ЗАДАЧІ 3](#_Toc508677010)

[1.1 Мета роботи 3](#_Toc508677011)

[1.2 Варіант та завдання лабораторної роботи 3](#_Toc508677012)

[2 ОПИС ПРОГРАМИ 3](#_Toc508677013)

[2.1 Опис коду 3](#_Toc508677014)

[2.2 Опис бібліотек та функцій 4](#_Toc508677015)

[ВИСНОВКИ 5](#_Toc508677016)

[ПЕРЕЛІК ПОСИЛАНЬ 5](#_Toc508677017)

[ДОДАТОК А (Відповіді на контрольні запитання) 5](#_Toc508677018)

[ДОДАТОК Б (Код програми) 9](#_Toc508677019)

[ДОДАТОК В (Результат роботи програми) 10](#_Toc508677020)

[ДОДАТОК Г (Блок-схема) 11](#_Toc508677021)

# ВСТУП

Даний звіт створено для детального ознайомлення з лабораторною роботою №3, її завданням, структурою, розробкою програмного коду мовою програмування C, у супроводі детального розбору поставлених вимог.

# 1 ПОСТАНОВКА ЗАДАЧІ

## 1.1 Мета роботи

Вивчити опис символьних даних та операцій над ними.

## 1.2 Варіант та завдання лабораторної роботи

Було виконано роботу за варіантом №24.

* Сформувати новий рядок, в якому всі слова старого рядка будуть йти в порядку збільшення довжини.

# 2 ОПИС ПРОГРАМИ

## 2.1 Опис коду

Програма починається з того, що користувачеві надається вибір між доступом до самої програми (кнопка «1»), очисткою консолі («8») та виходом з програми («9»). Реалізовано за допомогою циклу do-while, функцій strchr (перевіряє, чи входить введений символ в заданий масив) та getch (за її допомогою компьютер отримує символ одразу після натискання; користувачеві непотрібно підтверджувати свій вибір Enter`ом). Далі правильний символ надходить до switch-case, що розмежує кожну з дій окремою підпрограмою. Для того, щоб з одного ‘кейсу’ програма просто не переходила у інший – в кінці кожної з підпрограм ставиться break.

Очистка консолі виконується за допомогою команди system(“cls”); з бібліотеки <conio.h>, звідки й getch.

Вихід з програми реалізовано командою exit(0); з <stdlib.h>.

Основна програма починається з того, що користувач вводить число (у межах до 20), тобто, необхідну йому кількість слів. Звичайно, кожний ввід контролюється, щоб не виникло помилки. У цьому випадку – це ‘валідація’ натуральних чисел, що знаходиться в окремій функції. Для ‘валідації’ використовуються, окрім циклу while, ще такі функції, як strlen() (рахує кількість елементів у масиві), isdigit() (перевіряє числове значення(чи є елемент цифрою)) та atoi() (конвертує строковий тип до цілих чисел).

Далі вже користувач вводить самі слова (у двовимірний масив), що перевіряються за допомогою циклів do-while та for, функцій strlen та isalpha (повертає ненульове значення, коли досліджуваний елемент є літерою англійського алфавіту).

Коли всі слова введені, починається алгоритм ‘бульбашкового’ сортування слів по довжині (від меншого до більшого). Відсортований масив потрапляє на етап виведення його користувачеві на екран.

Після закінчення кожної підпрограми (окрім виходу з програми) відбувається рекурсивний повтор, що дозволяє ще раз обрати між заданими діями, натиснувши «1», «8» чи «9».

## 2.2 Опис бібліотек та функцій

Для початку, опишемо використані бібліотеки у програмі:

<stdio.h> - відповідає за ввод\вивід даних;

<stdlib.h> - відповідає за функції, що займаються виділенням пам'яті, контроль процесу виконання програми, перетворення типів та інші;

<string.h> - містить функції для роботи з нуль-термінованими рядками і різними функціями роботи з пам'яттю;

<ctype.h> - містить оголошення функцій для класифікації символів;

<conio.h> - застосовується для створення текстового інтерфейсу.

Опис функцій:

Програма починається з виконання функціїї int main(), в якій викликається валідація натуральних чисел:

1. int valid\_int()

# ВИСНОВКИ

Під час виконання лабораторної роботи були вивчені нюанси роботи із символьними даними у мові програмування С. Також ми ознайомились з різними корисними функціями, що описані в <string.h>.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Програмування мовою С. Методичні вказівки до виконання лабораторних робіт із кредитного модуля «Програмування-2. С.» для студентів спеціальності 113 «Прикладна математика» [Текст] / В. В. Громова. — К. : НТУУ «КПІ», 2016. —5- 49 с.

# ДОДАТОК А (Відповіді на контрольні запитання)

1. *Як описують рядки мовою С?*

Масиви символів, які завершуються нуль-символом '\0' (символ з ASCII-кодом 0).

1. *Функції для уведення та виведення символів.*

Для уведення символів можна використовувати таку функцію:

сhar getchar();

Або ще таку:

scanf(“%c”, &ch);

1. *Функції для введення та виведення рядків.*

Для введення рядка можна використовувати функцію scanf(). Проте, функцію scanf() призначено радше для отримання не рядка, а окремого слова. Якщо застосовувати формат "%s" для уведення, рядок уводиться до (але не включаючи) наступного порожнього символу, яким може бути пробіл, табуляція або символ нового рядка.

Для уведення рядка, включаючи пробіли, використовують таку функцію:

char\* gets(char\*);

1. *Функції перевірки символів.*

Для перевірки символів використовують функції, які повертають значення «істина» або «хиба». Прототипи цих функцій описано у файлі ctype.h бібліотеки стандартних функцій.

Функції з бібліотеки повертають значення «істина», якщо:

isalpha(c): c — символ алфавіту;

isupper(c): c — символ верхнього регістру;

islower(c): c — символ нижнього регістра;

isdigit(c): c — цифра від 0 до 9;

isxdigit(c): c — шістнадцяткова цифра;

isalnum(c): c — буква чи цифра;

isspace(c): c — символ пробілу, табуляції, переведення рядка.

1. *Функції, що реалізують операції з рядками.*

| Функція | Опис |
| --- | --- |
| char\* strcat(char\* s1, char\* s2) | приєднує s2 до s1, повертає s1 |
| char\* strncat(char\* s1, char\* s2, int n) | приєднує не більше за n символів s2 до s1, завершує рядок символом '\0', повертає s1 |
| char\* strсpy(char\* s1, char\* s2) | копіює s2 в s1, включаючи '\0', повертає s1 |
| char\* strncpy(char\* s1, char\* s2, int n) | копіює не більше за n символів s2 в s1, повертає s1 |
| intstrcmp(char\* s1, char\* s2) | порівнює s1 і s2, повертає значення 0, якщо рядки еквівалентні |
| intstrncmp(char\* s1, char\* s2, int n) | порівнює початкові n символів s1 і s2, повертає значення 0, якщо початкові n символів рядків еквівалентні |
| intstrlen(char\* s) | повертає кількість символів в s |
| char\* strset(char\* s, char c) | заповнює s символами, код яких дорівнює c, повертає вказівник на s |
| char\* strnset(char\* s, char c, int n) | замінює перші n символів s символами, код яких дорівнює c, повертає вказівник на s |

1. *З’єднання послідовностей символів.*

char\* strcat(char\* s1, char\* s2) – ця функція містьтся в бібліотеці <string.h>.

1. *Пошук першого входження в масив.*

int main() {

int first\_in = -1;

char string[6] = “Hello”;

for (int i = 0; i < 6; i++){

if(string[i] == ‘l’){

first\_in = i;

break;

}

}

printf(“%d”, first\_in);

}

1. *Порівняння рядків.*

| Функція | Опис |
| --- | --- |

|  |  |
| --- | --- |
| intstrcmp(char\* s1, char\* s2) | порівнює s1 і s2, повертає значення 0, якщо рядки еквівалентні |
| intstrncmp(char\* s1, char\* s2, int n) | порівнює початкові n символів s1 і s2, повертає значення 0, якщо початкові n символів рядків еквівалентні |

1. *Копіювання символів.*

**Синтаксис:**

#include <string.h>   
char \*strncpy (char \*destination, const char \*source, size\_t n);

char \*strcpy (char \*destination, const char \*source);

**Аргументи:**

destination – вказівник на рядок, до якого будуть скопійовані.  
source – вказівник на рядок.  
n – обмежувач довжини копіювання.

// strcpy (f\_array, s\_array);

1. *Визначення довжини рядка.*

#include <stdio.h>

#include <string.h>

char str[64] = “example”;

printf(“%d”, strlen(str)); // ВИВЕДЕ 7

1. *Скільки байтів буде виділено під розміщення масиву* a *в результаті такого оголошення:* char a[] = "ABCD";*?*

Буде виділено 4 байти.

1. *У чому різниця в оголошеннях* char\* b = "Array of char" і char b[] = "Array of char"*?*

Різниця тут у тому, що

char\* b = "Array of char"

буде розташовувати "Array of char" в пам’яті тільки для читання. b буде вказівником, через що робить будь-яку операцію запису на цю пам’ять незаконною.

char b[] = "Array of char"

Поміщає буквенний рядок в пам'ять для читання та копіює рядок до недавно виділеної пам'яті до стеку. Таким чином, ця дія

b[0] = ‘G’;

є законною.

1. *Наведіть приклад оголошення змінної, якій можна присвоїти результат обчислення виразу 'a'.*

char \*symbol\_a = (char\*)malloc(1\*sizeof(char));

1. *Чи можна в програмі* Turbo C *вказувати двосимвольні константи?*

Можна. ‘ab’, ‘vf’ – такий запис допустимий тільки якщо двосимвольну константу вказувати як *int*, а не як *char*, бо інакше ‘загубиться’ другий символ.

До прикладу, '\538' – це ‘+8’, бо ‘\53’ сприймається машиною як ‘+’, та залишається ‘8’. Аналогічно '\x6DQ' - це 'mQ'.

# ДОДАТОК Б (Код програми)

#include <stdio.h>

#include <string.h> // strlen(), strchr()

#include <stdlib.h> // atoi(), exit(0)

#include <ctype.h> // isdigit(), isalpha()

#include <conio.h> // system("cls"), getch()

#define word\_len 20

int valid\_int() {

char str[65];

int num;

while (1) {

int smb = 0;

scanf("%s", str); *//input of value*

int len = strlen(str);

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

if (isdigit(str[i]) == 0) {

++smb;

**break**;

}

}

if (smb == 0) {

num = atoi(str); *//convert str to int*

if (num > 20) printf("No bigger than 20!: ");

else **break**;

}

else printf("Need natural number!: ");

}

return num;

}

int main() {

int choice;

fflush(stdin);

printf("**\n**Press '1' to form and sort array.**\n**"\

"Press '8' to clear console.**\n**"\

"Press '9' to get out program.**\n**");

do {

choice = getch();

} while(!strchr("189", choice));

switch (choice) {

case '1': {

printf("**\n**Enter number of words: ");

int n = valid\_int();

char str[n][word\_len];

int nz = word\_len;

int \*iter = &nz;

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

do {

\*iter = 0;

printf("%d. yours symbols: ", i+1);

scanf("%s", &str[i]);

for (int j = 0; j < strlen(str[i]); ++j) {

if (!isalpha(str[i][j])) {

++\*iter;

**break**;

}

}

}while((\*iter != 0) || (strlen(str[i]) > word\_len));

}

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

for (int j = 1; j < n; ++j) {

if (strlen(str[j-1]) > strlen(str[j])) {

for (int k=0; k < word\_len; ++k) {

char switcher[1][ word\_len];

switcher[0][k] = str[j-1][k];

str[j-1][k] = str[j][k];

str[j][k] = switcher[0][k];

}

}

}

}

printf("sort\_array = {");

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

if (i != n-1) printf("%s, ", str[i]);

else printf("%s", str[i]);

}

printf("}**\n**");

**break**;

}

case '8': {

printf(" 7 ballov :)**\n**");

system("cls");

**break**;

}

case '9': exit(0); **break**;

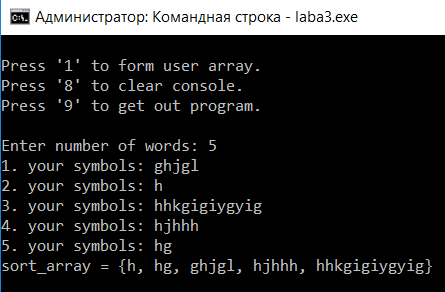
default: **break**;

}

return main();

}

# ДОДАТОК В (Результат роботи програми)



# ДОДАТОК Г (Блок-схема)

