Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет

ЗВІТ
ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 7
з навчальної дисципліни
“Базові методології та технології програмування”

ПРОГРАМНА РЕАЛІЗАЦІЯ ОБРОБЛЕННЯ МАСИВІВ ДАНИХ ТА СИМВОЛЬНОЇ ІНФОРМАЦІЇ

ЗАВДАННЯ ВИДАВ
доцент кафедри кібербезпеки
та програмного забезпечення
Доренський О. П.
https://github.com/odorenskyi/

ВИКОНАВ
студент академічної групи КБ-20
Ковальова Єва

ПЕРЕВІРИВ
ст. викладач кафедри кібербезпеки
та програмного забезпечення
Коваленко Анастасія Сергіївна

Кропивницький – 2025

ПРОГРАМНА РЕАЛІЗАЦІЯ ОБРОБЛЕННЯ МАСИВІВ ДАНИХ ТА СИМВОЛЬНОЇ ІНФОРМАЦІЇ ЗА СТАНДАРТОМ UNICODE

Мета роботи полягає у набутті грунтовних вмінь і практичних навичок синтезу алгоритмів оброблення масивів даних та символьної (текстової) інформації у кодуваннях UTF-8 і CP866, іх програмної реалізації мовою програмування мовою програмування C (ISO/IEC 9899:2018) задля реалізації програмних засобів у вільному кросплатформовому Code::Blocks IDE.

Аналіз задачі 7.1

Програма призначена для перевірки, чи міститься в введеному тексті слово "Кропивницький" або "Kropyvnytskyi", яке закінчується символом '|'. Вона підтримує введення як українською, так і англійською мовами та враховує можливі зайві символи в слові.

Основні етапи роботи програми:

1. **Налаштування кодування**
   - Програма встановлює підтримку UTF-8 у Windows-консолі, щоб коректно відображати українські символи.
2. **Зчитування тексту**
   - Користувач вводить рядок, що містить слова.
   - Програма шукає слово, яке закінчується на '|'.
3. **Перевірка відповідності**
   - Якщо знайдене слово точно збігається з "Кропивницький" або "Kropyvnytskyi", видається позитивний результат.
   - Інакше слово очищається від зайвих символів перед повторною перевіркою.
4. **Очищення слова**
   - Визначається, чи слово більше схоже на англійське або українське.
   - Відповідно до цього, видаляються зайві символи.
5. **Остаточна перевірка після очищення**
   - Якщо очищене слово збігається з "Кропивницький" або "Kropyvnytskyi", програма підтверджує відповідність.
   - Інакше повідомляє, що слово не збігається.
6. **Завершення роботи**
   - Очікує натискання Enter перед виходом.

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>
#include <locale.h>
#include <windows.h>
#include <ctype.h>

void cleanWordCyrillic(char *word) {
    char *dest = word;
    unsigned char c;

    while (*word) {
        c = (unsigned char)*word;
```

```
        if ((c == 0xD0 || c == 0xD1) && *(word + 1)) {
            *dest++ = *word++;
            *dest++ = *word;
        }
        word++;
    }
    *dest = '\0';
}

void cleanWordEnglish(char *word) {
    char *dest = word;
    unsigned char c;

    while (*word) {
        c = (unsigned char)*word;

        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z')) {
            *dest++ = *word;
        }
        word++;
    }
    *dest = '\0';
}

bool looksLikeEnglish(const char *word) {
    int englishChars = 0;
    int cyrillicChars = 0;
    unsigned char c;

    while (*word) {
        c = (unsigned char)*word;

        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z')) {
            englishChars++;
        } else if ((c == 0xD0 || c == 0xD1) && *(word + 1)) {
            cyrillicChars++;
            word++;
        }
        word++;
    }

    return englishChars > cyrillicChars;
}

int main() {
    SetConsoleOutputCP(65001);
    SetConsoleCP(65001);

    setlocale(LC_ALL, "uk_UA.UTF-8");

    printf("Автор програми: Ковальова Єва\n");
    printf("Програма перевіряє, чи є слово, яке закінчується символом '|',
словом 'Кропивницький'.\n");
    printf("Введення може бути українською ('Кропивницький') або англійською
('Kropyvnytskyi') мовою.\n\n");

    char sentence[1000];
    char targetWord[100] = "";
    char cleanedWord[100] = "";
    bool foundWordWithPipe = false;
```

```
    printf("Введіть речення (слово, яке треба перевірити, повинно
закінчуватися '|'): ");
    fgets(sentence, sizeof(sentence), stdin);

    sentence[strcspn(sentence, "\n")] = '\0';

    char *word = strtok(sentence, " \t\n");
    while (word != NULL) {
        int len = strlen(word);
        if (len > 0 && word[len - 1] == '|') {
            strncpy(targetWord, word, len - 1);
            targetWord[len - 1] = '\0';
            foundWordWithPipe = true;
            break;
        }
        word = strtok(NULL, " \t\n");
    }

    if (foundWordWithPipe) {
        printf("Знайдено слово з символом '|': %s\n", targetWord);

        if (strcmp(targetWord, "Кропивницький") == 0 || strcmp(targetWord,
"Kropyvnytskyi") == 0) {
            printf("Результат перевірки: Так, знайдене слово є
'Кропивницький'/'Kropyvnytskyi'.\n");
        } else {
            strcpy(cleanedWord, targetWord);

            bool isEnglish = looksLikeEnglish(targetWord);

            if (isEnglish) {
                cleanWordEnglish(cleanedWord);
                printf("Після очищення від спеціальних символів
(англійською): %s\n", cleanedWord);

                if (strcmp(cleanedWord, "Kropyvnytskyi") == 0) {
                    printf("Результат перевірки: Так, знайдене слово є
'Kropyvnytskyi'.\n");
                } else {
                    printf("Результат перевірки: Ні, знайдене слово не є
'Kropyvnytskyi'.\n");
                }
            } else {
                cleanWordCyrillic(cleanedWord);
                printf("Після очищення від спеціальних символів
(українською): %s\n", cleanedWord);

                if (strcmp(cleanedWord, "Кропивницький") == 0) {
                    printf("Результат перевірки: Так, знайдене слово є
'Кропивницький'.\n");
                } else {
                    printf("Результат перевірки: Ні, знайдене слово не є
'Кропивницький'.\n");
                }
            }
        }
    } else {
        printf("У реченні немає слова, яке закінчується символом '|'.\n");
    }

    printf("\nНатисніть Enter для завершення програми...");
    getchar();
```

```
    return 0;
}
```

Алгоритм дії програми

1. **Налаштування середовища**
   - Встановити кодування UTF-8 для коректного відображення українських символів у Windows-консолі.
   - Встановити локалізацію на українську мову.
   - Вивести інформацію про автора та опис програми.
2. **Отримання вхідного речення**
   - Користувач вводить речення, у якому слово для перевірки повинно закінчуватися символом |.
   - Видалити символ нового рядка \n, якщо він є.
3. **Пошук слова, яке закінчується символом |**
   - Розділити введене речення на слова за пробілами, табуляціями та новими рядками.
   - Перевірити кожне слово:
     - Якщо останній символ |, зберегти це слово (без |) як targetWord.
     - Завершити пошук, якщо слово знайдено.
4. **Перевірка знайденого слова**
   - Якщо targetWord дорівнює "Кропивницький" або "Kropyvnytskyi", вивести позитивний результат.
   - В іншому випадку перевірити, чи слово англійське або українське:
     - Порахувати кількість англійських та кириличних символів у слові.
     - Якщо більше англійських символів → англійське слово.
     - Інакше → українське слово.
5. **Очищення слова від зайвих символів**
   - Якщо слово англійське, залишити лише літери (a-z, A-Z).
   - Якщо слово українське, залишити лише кириличні символи.
6. **Фінальна перевірка**
   - Якщо очищене слово дорівнює "Кропивницький" або "Kropyvnytskyi", вивести позитивний результат.
   - В іншому випадку — вивести, що слово не є "Кропивницький".
7. **Завершення програми**
   - Якщо слово з | не було знайдено, вивести відповідне повідомлення.
   - Очікувати натискання Enter перед завершенням.

Аналіз задачі 7.2

Ця програма автоматично генерує масив із 20 натуральних чисел, знаходить найменший елемент і замінює його на суму всіх чисел у масиві. Вона використовує випадкову генерацію чисел у заданому діапазоні та відображає як початковий, так і змінений масив.

Основні етапи роботи програми

1. **Налаштування кодування та локалізації**
   - Встановлення кодування UTF-8 для коректного відображення українських символів у Windows-консолі.
2. **Генерація випадкового масиву**
   - Випадковий генератор ініціалізується поточним часом.
   - Масив заповнюється випадковими натуральними числами в діапазоні [1, 100].
   - Вхідний масив виводиться на екран у зручному форматі.

3. **Пошук найменшого елемента**
    - Програма проходить масив, знаходить мінімальне значення та його індекс.
    - Паралельно обчислюється сума всіх елементів.
4. **Заміна найменшого елемента на суму всіх чисел**
    - Найменший елемент у масиві замінюється на загальну суму чисел.
5. **Вивід оновленого масиву**
    - Масив після модифікації виводиться у тому ж форматі.
6. **Завершення роботи**
    - Очікується натискання Enter перед закриттям програми.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <locale.h>
#include <windows.h>

#define ARRAY_SIZE 20

int main() {
    SetConsoleOutputCP(65001);
    SetConsoleCP(65001);

    setlocale(LC_ALL, "uk_UA.UTF-8");

    printf("Автор програми: Ковальова Єва\n");
    printf("Програма замінює найменший елемент масиву на суму всіх
елементів.\n");
    printf("Масив з %d натуральних чисел генерується автоматично.\n\n",
ARRAY_SIZE);

    int minRange = 1;
    int maxRange = 100;

    srand((unsigned int)time(NULL));

    int array[ARRAY_SIZE];

    printf("Вхідний масив:\n[ ");
    for (int i = 0; i < ARRAY_SIZE; i++) {
        array[i] = minRange + rand() % (maxRange - minRange + 1);
        printf("%d", array[i]);
        if (i < ARRAY_SIZE - 1) {
            printf(", ");
        }
        if ((i + 1) % 10 == 0) {
            printf("\n  ");
        }
    }
    printf(" ]\n\n");

    int minValue = array[0];
    int minIndex = 0;
    int sum = array[0];

    for (int i = 1; i < ARRAY_SIZE; i++) {
        sum += array[i];

        if (array[i] < minValue) {
```

```
            minValue = array[i];
            minIndex = i;
        }
    }

    printf("Найменший елемент: %d (індекс %d)\n", minValue, minIndex);
    printf("Сума всіх елементів: %d\n\n", sum);

    array[minIndex] = sum;

    printf("Вихідний масив (після заміни найменшого елемента на суму):\n[ ");
    for (int i = 0; i < ARRAY_SIZE; i++) {
        printf("%d", array[i]);
        if (i < ARRAY_SIZE - 1) {
            printf(", ");
        }
        if ((i + 1) % 10 == 0) {
            printf("\n  ");
        }
    }
    printf(" ]\n");

    printf("\nНатисніть Enter для завершення програми...");
    getchar();

    return 0;
}
```

Алгоритм дії програми

1. **Налаштування середовища**
   - Встановити кодування UTF-8 для коректного відображення українських символів у консолі.
   - Встановити локалізацію на українську мову.
2. **Оголошення змінних**
   - Визначити розмір масиву (ARRAY_SIZE = 20).
   - Визначити діапазон випадкових чисел (minRange = 1, maxRange = 100).
   - Оголосити масив array[ARRAY_SIZE].
3. **Генерація випадкових чисел**
   - Ініціалізувати генератор випадкових чисел (srand(time(NULL))).
   - Заповнити масив випадковими числами в заданому діапазоні [1, 100].
   - Вивести початковий масив у зручному форматі.
4. **Пошук найменшого елемента та підрахунок суми**
   - Встановити перший елемент як мінімальний (minValue = array[0]).
   - Ініціалізувати змінну sum = array[0].
   - Пройти циклом по всьому масиву:
     - Додавати кожен елемент до sum.
     - Якщо знайдено менше число, оновити minValue та minIndex.
5. **Заміна найменшого елемента**
   - Встановити array[minIndex] = sum.
6. **Вивід оновленого масиву**
   - Вивести змінений масив після заміни найменшого елемента.
7. **Завершення програми**
   - Очікувати натискання Enter перед завершенням.

1. Формує базові навички роботи з масивами даних у мові С.
2. Дозволяє освоїти кодування UTF-8 і CP866.
3. Покращує розуміння стандарту Unicode.
4. Розширює знання щодо ISO/IEC 9899:2018 (стандарт С).
5. Дає практичний досвід використання Code::Blocks IDE.
6. Навчає основам текстової обробки в С.
7. Покращує розуміння алгоритмів роботи з символами.
8. Дозволяє працювати з ASCII-таблицями.
9. Дає змогу обробляти текстові файли в різних кодуваннях.
10. Ознайомлює з проблемами несумісності символів у різних системах.
11. Навчає працювати з вхідними та вихідними потоками даних.
12. Формує навички конвертації між різними форматами тексту.
13. Дозволяє закріпити знання про динамічні масиви в С.
14. Розширює розуміння структурованого програмування.
15. Покращує навички роботи з GitHub для спільної розробки.
16. Ознайомлює з BSD-ліцензією та її особливостями.
17. Вчить працювати з онлайн-ресурсами ASCII-кодів.
18. Дає змогу використовувати редактори коду для тестування.
19. Підвищує ефективність використання командної строки в Linux.
20. Підсилює навички налагодження програм у Code::Blocks.
21. Дає практику написання програм для кросплатформних середовищ.
22. Навчає ефективній роботі з таблицями символів.
23. Показує особливості роботи з кирилицею та латиницею в різних кодуваннях.
24. Сприяє розвитку алгоритмічного мислення.
25. Навчає роботі з текстовими потоками в С.
26. Показує методи перевірки коректності зчитування символів.
27. Дозволяє уникнути поширених помилок при роботі з текстовими масивами.
28. Пояснює різницю між однобайтовими та багатобайтовими кодуваннями.
29. Розвиває вміння тестувати власні програмні рішення.
30. Формує навички оптимізації алгоритмів обробки символів.
31. Покращує навички роботи з середовищем Code::Blocks.
32. Навчає використовувати Unicode-коди для різних мов.
33. Дає практичне розуміння роботи зі стандартними бібліотеками С.
34. Ознайомлює з особливостями обробки різних мовних символів.
35. Дає досвід роботи з основними функціями обробки тексту (fgets, printf, scanf).
36. Вчить знаходити та виправляти проблеми з кодуванням.
37. Покращує вміння використовувати довідкові матеріали ASCII та Unicode.
38. Розширює можливості роботи з символами через умовні оператори.
39. Дозволяє навчитися змінювати кодування файлів у різних середовищах.
40. Навчає створювати тести для перевірки правильності обробки символів.
41. Ознайомлює з внутрішнім представленням символів у пам’яті.
42. Формує розуміння відмінностей між 8-бітними та 16-бітними кодуваннями.
43. Дозволяє працювати з реальними даними для аналізу кодувань.
44. Навчає адаптувати програми для різних систем і мов.
45. Підсилює знання про роботу з буферами введення/виведення.
46. Навчає обробці винятків, пов’язаних із некоректними кодуваннями.
47. Розвиває навички документування коду та написання коментарів.
48. Формує розуміння роботи з багатомовними текстовими файлами.

49.Навчає використовувати правильні символи для форматування тексту.
50.Дає змогу застосувати теоретичні знання в реальних завданнях.

- **git init**: Ініціалізує новий Git репозиторій у поточній директорії.
- **git add (git add .)**: Додає зміни у файлах до індексу (staging area) для подальшого коміту. git add . додає всі зміни в поточній директорії та її піддиректоріях.
- **git commit (git commit -m "текст коміту")**: Зберігає знімки змін з індексу в історію репозиторію. Опція -m дозволяє додати повідомлення до коміту.
- **git remote add (git remote add origin [видалено електронну адресу]:username/BMTP-LAB7-прізвище.git)**: Додає віддалений репозиторій (наприклад, на GitHub) до локального репозиторію. "origin" - це ім'я віддаленого репозиторію.
- **git push (git push origin master)**: Відправляє локальні коміти у віддалений репозиторій. origin - це ім'я віддаленого репозиторію, master - ім'я гілки.