НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ЗВІТ

про виконання лабораторної роботи №3

Виконав: Климок Н.І.
студент групи ПЗ - 16

_____
(дата виконання)

Львів - 2022

Тема. Зміст та роль тестування у життєвому циклі розробки програмного забезпечення.

Мета. Вивчити основні методи та принципи тестування на прикладі створення власної програми.

ТЕОРЕТИЧНІ ВІДОМОСТІ

6. Гранично-допустимі випробування – тестування, що проводиться для визначення залежностей мід гранично допустимими значеннями функції чи об'єкта і режимом експлуатації.
Приклад: програма-анкета для друзів, ввід імені користувача (від 3 до 50 символів)

1. *Ввід пустого рядка*
2. *Ввід рядка з 50 символів*
3. *Ввід 3 символів*

26. Приклади нефункціонального тестування, що стосується атрибутів якості ПЗ:

- *Тестування веб-сервісу велетенською кількістю одночасно активних користувачів (стресове тестування) – якість оптимізацій та алгоритмів*
- *Тестування веб-сервісу після збереження великої кількості інформації до бази даних (об'ємне тестування) – якість конфігурації сховища*
- *Тестування зручності користування – якість інтерфейсу та UX.*

34. Життєвий цикл змінної визначає створення, ініціалізацію, використання і, ймовірно, знищення змінної. Відхилення від сценарію життєвого циклу змінної має занепокоїти тестувальника.

ПОСТАНОВКА ЗАВДАННЯ

1. Провести функціональне тестування програми для роботи з динамічними списками (розробленої у ході виконання лабораторної роботи з дисципліни «Основи програмування»), використавши такі тести:

1. димовий тест,
2. тест нормальних умов,
3. тест екстремальних(граничних) умов,
4. тест виняткових умов.

1.1. Результати тестів 1, 2, 3 та 4 оформити у вигляді таблиці (за зразок взяти таблицю 2). У таблиці інформація заноситься в зрозумілій формі, достатньо повно, щоб не вимагалося додаткових (усних) пояснень. Для представлення *тестових даних* та *фактичних результатів* бажано використовувати посилання на зображення екранів чи відповідні файли. В *очікуваних результатах* зафіксувати, яка функція була протестована та відповідно, що є очікуваним результатом. Якщо помилки не знайдено у *Тип звіту* ставимо знак -. Примітки використовуємо для додаткових пояснень.

1.2. Знайдені дефекти виправити. Провести повторне тестування, результати якого оформити окремою таблицею.

2. Окремо провести тестування логічної схеми програми (головна програма та дві функції-підпрограми). Для цього зобразити її у вигляді графів управління. Вершини графів пронумерувати. Записати усі отримані маршрути роботи програми, використовуючи номери вершин.

2.1. Для кожного маршруту провести тестування і записати результати тестування у звітній таблиці.

2.2. Знайдені дефекти виправити. Провести повторне тестування, результати якого оформити окремою таблицею.

Виконання завдання

1. Виконаємо функціональне тестування програми для роботи з динамічними списками.

1.1 Димовий тест

1.2 Тест нормальних умов

1.3 Тест екстремальних (граничних) умов

1.4 Тест виняткових умов

№ з/п	Тестові дані	Фактичні результати	Очікувані результати	Ступінь критичності	Тип звіту	Примітки
Тест Нормальних Умов						
1	Запуск виведення даних з файлу	Коректне виведення усіх елементів	Коректне виведення усіх елементів	-	-	Рис.1
2	Видалення елементів, що підходять умові	Видалення елементів і виведення усіх елементів	Видалення елементів і виведення усіх елементів	-	-	Рис.2
Тест Граничних Умов						
1	Запуск з пустим вхідних файлом	Помилка сегментації	Створення пустого списку	Некритична	Помилка кодування	Рис.3
2	Запуск з 255 символами	Штатне функціонування програми	Штатне функціонування програми	-	-	Рис.4
Тест Виняткових Ситуацій						
1	Запуск із неіснуючим файлом	Помилка сегментації	Вивід помилки користувачу	Некритична	Помилка кодування	Рис.5

Рис.1:

```
nazariiklymok@Nazariis-MacBook-Pro lab3 % ./a.out
George Orwell |  1984 book | 1948 | 240 | 25.000000
George Orwell |  Animal farm | 1945 | 112 | 16.000000
Taras Shevchenko |  Kobzar book | 1840 | 120 | 23.000000
Artur Schopenhauer |  Life wisdom | 1851 | 459 | 15.000000
What would you like to do? (i - insert node, d - delete node, q - quit) :
```

Рис. 2:

```
What would you like to do? (i - insert node, d - delete node, q - quit) : d
George Orwell |  1984 book | 1948 | 240 | 25.000000
George Orwell |  Animal farm | 1945 | 112 | 16.000000
Taras Shevchenko |  Kobzar book | 1840 | 120 | 23.000000
Artur Schopenhauer |  Life wisdom | 1851 | 459 | 15.000000
What would you like to do? (i - insert node, d - delete node, q - quit) :
```

Рис. 3:

```
nazariiklymok@Nazariis-MacBook-Pro lab3 % ./a.out
zsh: segmentation fault  ./a.out
```

Рис. 4:

```
nazariiklymok@Nazariis-MacBook-Pro lab3 % ./a.out
George Orwellaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa |  1984
 book | 1948 | 240 | 25.000000
George Orwell |  Animal farm | 1945 | 112 | 16.000000
Taras Shevchenko |  Kobzar book | 1840 | 120 | 23.000000
Artur Schopenhauer |  Life wisdom | 1851 | 459 | 15.000000
```

Рис. 5:

```
nazariiklymok@Nazariis-MacBook-Pro lab3 % ./a.out
zsh: segmentation fault  ./a.out
```

Виправлення помилок

Виправлення помилки №1, гранична умова:

```
char *input = read_file("input.txt");
if (!strlen(input))
{
   printf("file is empty, exiting...\n");
   exit(1);
}
```

Виправлення помилки №1, виняткова ситуація:

```
FILE *fp = fopen(filename, "r");
if (!fp)
{
  printf("file does not exist. Exiting...\n");
  exit(1);
}
```

Повторне тестування

№ з/п	Тестові дані	Фактичні результати	Очікувані результати	Ступінь критичності	Тип звіту	Примітки
Тест Нормальних Умов						
1	Запуск виведення даних з файлу	Коректне виведення усіх елементів	Коректне виведення усіх елементів	-	-	-
2	Видалення елементів, що підходять умові	Видалення елементів і виведення елементів	Видалення елементів і виведення елементів	-	-	-
Тест Граничних Умов						
1	Запуск з пустим вхідних файлом	Створення пустого списку	Створення пустого списку	-	-	Рис. 1
2	Запуск з 255 символами	Штатне функціонування програми	Штатне функціонування програми	-	-	-
Тест Виняткових Ситуацій						
1	Запуск із неіснуючим файлом	Вивід помилки користувачу	Вивід помилки користувачу	-	-	Рис. 2

Рис. 1:

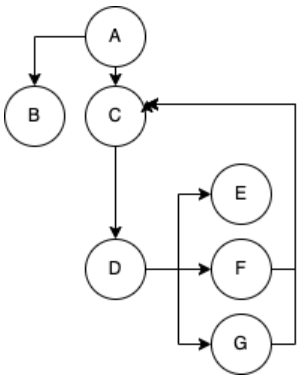
```
nazariiklymok@Nazariis-MacBook-Pro lab3 % ./a.out
file is empty, exiting...
```

Рис. 2:

```
nazariiklymok@Nazariis-MacBook-Pro lab3 % ./a.out
file does not exist. Exiting...
```

Структурне тестування

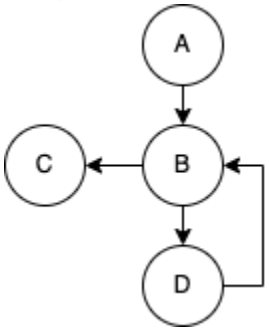
Головна функція



Обхід:

1) A->B
2) A->C->D->E
3) A->C->D->F
4) A->C->D->G

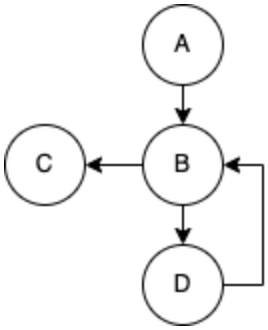
Функція-підпрограма print_books():



Обхід:

1) A->B->C
2) A->B->D->B->C

Функція-підпрограма get_choice():



Обхід:

1) A->B->C
2) A->B->D->B->C

№ з/п	Тестові дані	Фактичні результати	Очікувані результати	Ступінь критичності	Тип звіту	Примітки
Функція main						
1	Ввід пустого/неіснуючого файлу	Повідомлення про помилку	Повідомлення про помилку	-	-	-
2	Вихід з програми	Вихід	Вихід	-	-	-
3	Видалення ел-ів, що підходять умові	Видалення елементів	Видалення елементів	-	-	-
4	Додавання елемента	Додавання елемента	Додавання елемента	-	-	-
Функція print_books()						
1	Вивід пустого списку	Миттєвий вихід із функції	Миттєвий вихід із функції	-	-	-
2	Вивід наповненого списку	Вивід усіх елементів	Вивід усіх елементів	-	-	-
Функція get_choice()						
1	Вибір існуючої опції	Виконання функції	Виконання функції	-	-	-
2	Вибір неіснуючої опції	Повторний запит опції	Повторний запит опції	-	-	-

Висновок

У цій лабораторній роботі я вивчив основні методи та принципи тестування на прикладі створення власної програми.