

Міністерство Освіти України
Державний університет Телекомунікацій

Лабораторна робота 10
Тестування та управління змінами
програмного забезпечення

Підготував: студент групи ПД-21
Гапей Максим Юрійович
Перевірила: викладач
Поперешняк С.В.

Мета – набути навичок в організації тестування та управлінні змінами програмного забезпечення.

Завдання 1.

Сценарій тестування “2D гра в стилі Evolution” : Тестування механіки

Завдання 2.

План тестування:

1. Створення нового запису

1.1. Налаштування вхідних параметрів

1.1.1. Перевірка полів введення

1.1.2. Перевірка кнопок натискання

1.1.3. Перевірка бігунків

1.1.4. Перевірка вибору локацій на карті

1.2. Перевірка наявності створеного запису

1.3. Відтворення створеного запису

2. Зовнішні і внутрішні чинники навколишнього середовища

2.2. Перевірка відтворення змін зовнішніх чинників

2.2.1. Тривалість дня відповідно до заданого часу

2.2.2. Зміна дня і ночі

2.2.3. Зміна погоди

2.2.4. Зміна пори року

2.2.5. Тривалість одного сезону

2.3. Зміна кількості необхідних ресурсів для існування ботів

2.3.1. Кількість води

2.3.2. Кількість рослинності

2.3.3. Кількість м'яса

3. Тестування штучного інтелекту

3.1. Перевірка відповідності їх характеристик поточно-заданим

3.1.1. Стартова кількість ботів відповідно до заданої

3.1.2. Тип ботів

3.1.3. Рівень розвитку

3.2. Перевірка наявності мутагену

3.3. Перевірка змін статистичних даних

3.3.1. Перевірка гнучкості алгоритму побудування діаграм у різних ситуаціях

3.3.2. Перевірка впливу дій кожного із ботів на оточуюче середовище

3.4. Зміна поточних параметрів під час тестування

3.4.1. Перевірка змін харчів

2.4.2. Перевірка змін у тривалості дня

2.4.3. Перевірка змін у тривалості одного сезону

2.4.4. Перевірка зміни типу ботів

4. Тестування бази даних за різних умов

4.1. Перевірка збереження даних при Нормальному виході з програми

4.2. Перевірка збереження даних при Критичному виході з програми

4.3. Збереження усіх необхідних точок для подальшого відтворення

4.4. Відтворення збереженого запису після тестування III

4.5. Перевірка умов збереження запису в критичних точках (save point)

Завдання 3.

Специфікація тестового варіанта №1	
Назва взаємодіючих класів: Стартове меню, меню налаштувань	Назва тесту: CreateTest
Опис тесту: перевірка правильності роботи процедура створення нового запису	
Початкові умови: Немає	
Очікуваний результат: створення нового запису відповідно до заданих параметрів	
Отриманий результат: користувач створить особистий запис і зможе його завантажувати в майбутньому	

Специфікація тестового варіанта №2	
Назва взаємодіючих класів: Логіка	Назва тесту: LogicTest
Опис тесту: перевірка очікуваної зміни внутрішніх і зовнішніх чинників	
Початкові умови: якщо є існуючий запис	
Очікуваний результат: правильне відтворення і зміни відповідних чинників	
Отриманий результат: зміни у віртуальному світі відбулися відповідно очікуванням користувача	

Специфікація тестового варіанта №3	
Назва взаємодіючих класів: Тестування штучного інтелекту	Назва тесту: AIconfigTest
Опис тесту: тест перевіряє правильність взаємодії ботів у віртуальному світі. їх зміни у поведінці та відтворення цих змін у статистичній базі даних та на діаграмі	
Початкові умови: відтворення існуючого запису	
Очікуваний результат: боти повинні змінювати свої характеристики в продовж часу, і ці зміни повинні відображатися на 3D діаграмі	
Отриманий результат: відбулися відповідні зміни у поведінці і характеристики ботів, а також поточне відтворення цих змін на статистичній діаграмі	

Завдання 4.

4.Розробити набір тестових даних для наступних компонентів (використовуючи Activity diagram):

- програма сортування масивів цілих чисел;
- програма, яка вираховує кількість символів (відмінних від пропусків) в текстових строках;
- програма, яка перевіряє текстові строки і замінює послідовності пропусків одним пропуском, а там де один пропуск – змінює його на символ %.

Тестовий набір даних для програми сортування масивів цілих чисел:

Специфікація тестового варіанта	
Назва взаємодіючих класів: Сортування масиву цілих чисел	Назва тесту: SortTest
Опис тесту: тест перевіряє правильність сортування масиву при введенні цілих чисел, а також реакція програми на введення раціональних чисел	
Початкові умови: введення з клавіатури ряду чисел	
Очікуваний результат: програма правильно відсортовує масив. У випадку введення нецілих чисел - видає попередження щодо введених раціональних чисел	
Отриманий результат: відбулися відповідні зміни у поведінці і характеристиці ботів, а також поточне відтворення цих змін на статистичній діаграмі	

Тестовий набір даних для програми, яка вираховує кількість символів (відмінних від пропусків) в текстових строках:

Специфікація тестового варіанта	
Назва взаємодіючих класів: Обрахування кількості символів в заданій текстовій строці	Назва тесту: SymbolTest
Опис тесту: перевірка правильності виведення кількості символів в рядку (за умови що всі пропуски ігноруються як символ)	
Початкові умови: введення з клавіатури текстової строки	
Очікуваний результат: програма правильно визначає кількість символів в рядку	
Отриманий результат: на прикладі рядка “I test code” програма видала результат 9 символів, що являється правильним підрахуванням символів	

Тестовий набір даних для програми, яка перевіряє текстові строки і замінює послідовності пропусків одним пропуском, а там де один пропуск – змінює його на символ %.

Специфікація тестового варіанта	
Назва взаємодіючих класів: Заміна ряду пропусків в рядку на один, а один пропуск на символ %	Назва тесту: ReplaceTest
Опис тесту: перевірка правильності заміни ряду пропусків на один пропуск, а один пропуск на відповідний символ	
Початкові умови: введення з клавіатури текстової строки	
Очікуваний результат: програма правильно замінює відповідні символи до заданих умов	
Отриманий результат: на прикладі рядка “I test code” програма видала результат “I%test code”	

Завдання 5.

1. Сортування масиву цілих чисел:

*Швидке сортування Хоара.

Алгоритм:

```
#include <iostream>
#include <iomanip>

void scanArray(int* arr, const int N) {
    for (int i = 0; i < N; i++) {
        std::cout << "a" << i + 1 << ": ";
        std::cin >> arr[i];
    }
    std::cout << std::endl;
}

void printArray(int* arr, const int N) {
    for (int i = 0; i < N; i++) {
        std::cout << std::setw(5) << arr[i];
    }
    std::cout << std::endl;
}

void swapping(int& a, int& b) {
    int temp;

    temp = a;
    a = b;
    b = temp;
}
```

```

}

void quickSort(int* arr, int left, int right)
{
    int l = left;
    int r = right;

    int mid = arr[(l + r) / 2];

    do {
        while (arr[l] < mid) l++;
        while (arr[r] > mid) r--;

        if (l <= r) {
            swapping(arr[l], arr[r]);
            l++;
            r--;
        }
    } while (l < r);

    if (left < r) quickSort(arr, left, r);
    if (l < right) quickSort(arr, l, right);
}

int main() {
    int size;

    std::cout << "Enter size of array: ";
    std::cin >> size;

    int* arr = new int[size];

    scanArray(arr, size);
    printArray(arr, size);

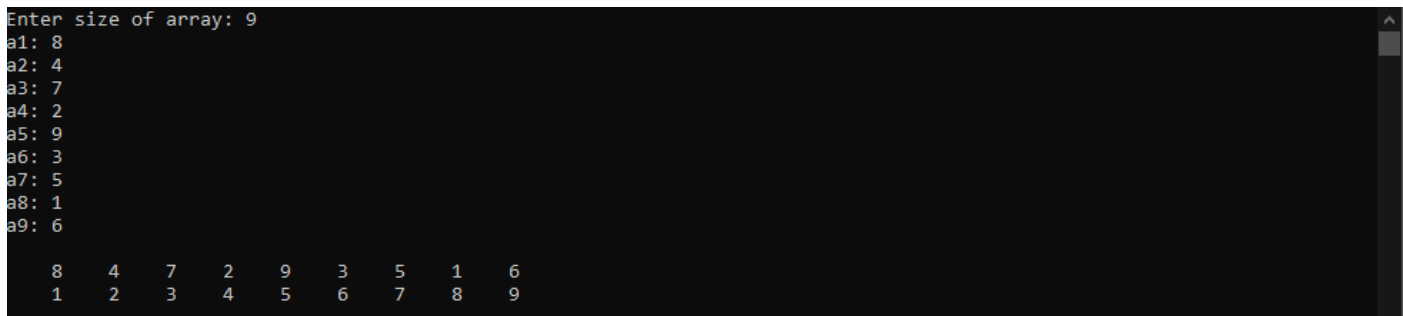
    quickSort(arr, 0, size - 1);

    printArray(arr, size);

    return 0;
}

```

Результат виконання програми:



```

Enter size of array: 9
a1: 8
a2: 4
a3: 7
a4: 2
a5: 9
a6: 3
a7: 5
a8: 1
a9: 6

8 4 7 2 9 3 5 1 6
1 2 3 4 5 6 7 8 9

```

2. Підрахування кількості символів у рядку

Алгоритм:

```

#include <iostream>
#include <string>

unsigned int SymbolNumber(std::string str)

```

```

{
    unsigned int flag = 0;
    for (unsigned int i = 0; i < str.length(); i++)
    {
        if (str[i] != ' ')
        {
            flag++;
        }
        else continue;
    }
    return flag;
}

void InputString(std::string& str)
{
    getline(std::cin, str);
}

int main()
{
    std::string str;

    std::cout << "Enter string: ";
    InputString(str);

    std::cout << "Symbols in string: " << SymbolNumber(str) << std::endl;

    return 0;
}

```

Результат виконання програми:

```

Enter string: I test code
Symbols in string: 9

```

3. Заміна ряду пробілів одним пробілом, а один пробіл символом %:

Алгоритм:

```

#include <iostream>
#include <string>

std::string ReplaceSymbol(std::string str)
{
    unsigned int flag = 0;
    std::string new_str;
    for (unsigned int i = 0; i < str.length(); i++)
    {
        if (str[i] == ' ')
        {
            flag++;
        }
        else
        {
            if (flag < 1)
            {
                new_str += str[i];
            }
            else if (flag == 1)
            {
                new_str += '%';
                flag = 0;
                i--;
            }
        }
    }
}

```



```

        else
        {
            new_str += ' ';
            flag = 0;
            i--;
        }
    }
    return new_str;
}

void InputString(std::string& str)
{
    getline(std::cin, str);
}

int main()
{
    std::string str;


    std::cout << "Enter string: ";
    InputString(str);

    std::cout << "New string: " << ReplaceSymbol(str) << std::endl;

    return 0;
}

```

Результат виконання програми:



```

Enter string: I test  code in  visual studio!
New string: I%test code%in visual%studio!

```

Висновок: набули навичок в організації тестування та управлінні змінами програмного забезпечення.