

**Міністерство освіти і науки України**  
**Львівський національний університет імені Івана Франка**  
**Факультет прикладної математики та інформатики**

Кафедра програмування

Лабораторна робота №9  
**БІТОВА МНОЖИНА**  
з курсу “Алгоритми та структури даних”

Виконав:  
Студент групи ПМІ-12  
Бенько Володимир Сергійович

Львів – 2023

## Бітова множина

Множина – це неупорядкована сукупність деяких об'єктів, які називають елементами множини і над якими не визначено жодного відношення.

Бітова множина – спосіб комп'ютерного подання множин з зарання визначеною множиною можливих елементів (універсальною множиною). В цьому випадку елементи універсуму нумерують, а множину відображають бітовим вектором де належність елемента з номером  $N$  множині відображається значенням  $N$ -ного біта.

Клас `CharSet` – множина символів реалізована Бітовою множиною, де в ролі універсуму виступає ASCII таблиця. В ролі бітового вектора виступає масив з 8 змінних цілочисельного типу розміром 32 біти кожна (256 в сумі). Клас містить набір публічних методів для операцій над множинами:

- `CharSet(string s)` - конструктор, приймає рядок `s` та додає в множину набір символів, які містяться в рядку.
- `Add(char a)` - додає символ `a` в множину.
- `Delete(char a)` - видаляє символ `a` з множини.
- `Test(char a)`: перевіряє чи символ `a` міститься в множині.
- `Association(CharSet& set)` - повертає новий об'єкт типу `CharSet`, який відповідає об'єднанню двох множин - поточного та `set`.
- `Intersection(CharSet& set)` - повертає новий об'єкт типу `CharSet`, який відповідає перетину двох множин - поточного та `set`.
- `Difference(CharSet& set)` - повертає новий об'єкт типу `CharSet`, який відповідає різниці двох множин - поточного та `set`.
- `operator==(CharSet& t1, CharSet& t2)` - оператор порівняння, який повертає `true`, якщо дві множини `t1` та `t2` рівні (містять однакові елементи), інакше повертає `false`.
- `Output(ostream& os)` - виводить множини у потік `os`.

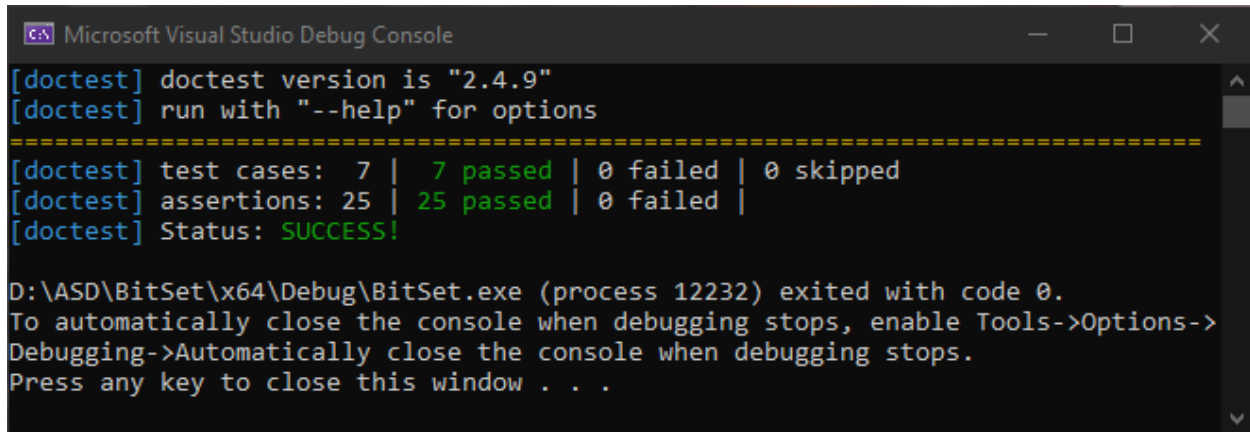
### Складність роботи методів:

- `CharSet`:  $O(n)$
- `Add`:  $O(1)$
- `Delete`:  $O(1)$
- `Test`:  $O(1)$
- `Association`:  $O(1)$
- `Intersection`:  $O(1)$
- `Difference`:  $O(1)$
- `operator==`:  $O(1)$
- `Output`:  $O(n)$

Просторова складність бітової множини:  $O(1)$

### Приклад:

Щоб переконатись, що всі функції працюють правильно, в програмі написані юніт-тести. Усі вони проходять успішно:

The image shows a screenshot of the Microsoft Visual Studio Debug Console window. The title bar reads "Microsoft Visual Studio Debug Console". The console output shows the following: [doctest] doctest version is "2.4.9", [doctest] run with "--help" for options, followed by a separator line of dashes. Then, it displays test results: test cases: 7 | 7 passed | 0 failed | 0 skipped, assertions: 25 | 25 passed | 0 failed |, and Status: SUCCESS!. At the bottom, it states: D:\ASD\BitSet\x64\Debug\BitSet.exe (process 12232) exited with code 0. To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops. Press any key to close this window . . .

```
Microsoft Visual Studio Debug Console
[doctest] doctest version is "2.4.9"
[doctest] run with "--help" for options
=====
[doctest] test cases:  7 |  7 passed | 0 failed | 0 skipped
[doctest] assertions: 25 | 25 passed | 0 failed |
[doctest] Status: SUCCESS!

D:\ASD\BitSet\x64\Debug\BitSet.exe (process 12232) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->
Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

### Висновок:

Бітова множина – структура даних, яка дозволяє досить ефективну роботу з множинами. Використання бітової множини дозволяє зберігати велику кількість даних на досить малому обсязі пам'яті, а також забезпечує швидке та ефективне виконання операцій з цими даними за допомогою бітових операцій.