Обробка послідовностей

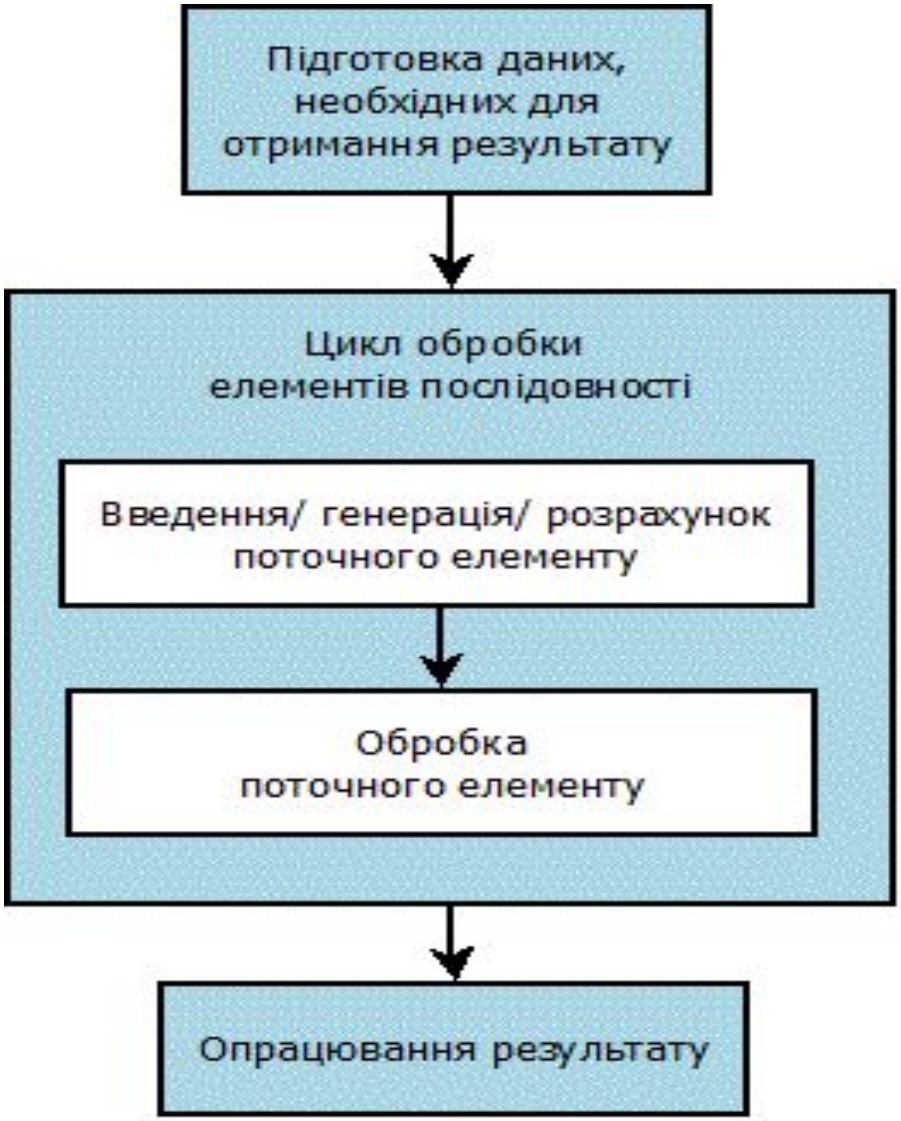
Загальна схема обробки.

Приклади алгоритмів обробки послідовностей

Нюанси обробки послідовностей

- Для представлення послідовності в більшості випадків достатньо використовувати **лише одну змінну для зберігання поточного елементу**
- **Обробка даних практично завжди суміщена з веденням/ генерацією/ розрахунком даних**.
- Схему обробки послідовності можна в подальшому проектувати на обробку цифр числа, одновимірного масиву тощо

Загальна схема обробки послідовності



1. Перед циклом обробки виконується підготовка даних, необхідних для формування результату. В залежності від задачі блок підготовки може бути відсутнім.
2. На кожному кроці циклу обробки відбувається формування поточного елементу та його обробка.
3. Після завершення циклу обробки відбувається опрацювання результату (виведення, перевірка+виведення, використання для вирішення інших задач тощо)

Сума елементів послідовності довжиною N (використовується метод накопичення)

```
S = 0;// підготовка початкового значення для розрахунку суми


for (int i = 1; i <= N; i++) //цикл для обробки елементів послідовності
{
    cout << "a"<<i<<" = ";
    cin >> a;          //введення поточного елементу
    S += a;        //обробка поточного елементу – додавання його до суми
}


cout << "Sum = " << S <<endl;//опрацювання результату – виведення значення суми
```

Кількість нулів в послідовності довжиною N (використовується метод накопичення)

```
k = 0; // підготовка початкового значення для розрахунку кількості

for (int i = 1; i <= N; i++) //цикл для обробки елементів послідовності
{
    cout << "a"<<i<<" = ";
    cin >> a;           //введення поточного елементу
    if (a == 0)         //обробка поточного елементу – перевірка елементу
та
        k++;        //зміна значення кількості k
}

cout << "Zero numbers = " << k <<endl;//опрацювання результату – виведення
кількості
```

Середнє арифметичне послідовності довжиною N (використовується метод накопичення)

```
Avg = 0; // підготовка початкового значення для розрахунку середнього арифметичного


for (int i = 1; i <= N; i++) //цикл для обробки елементів послідовності:
{
    cout << "a"<<i<<" = ";
    cin >> a;        //введення поточного елементу
    Avg += a;        //обробка поточного елементу – формування суми елементів
}


cout << "Avg =" << Avg/N <<endl; //опрацювання результату – розрахунок та виведення середнього арифметичного
```

Середнє арифметичне ненульових елементів послідовності довжиною N (використовується метод накопичення)

```
Avg = 0; // підготовка початкових значень для розрахунку середнього арифметичного
k = 0;   // ненульових елементів: для розрахунку необхідно знайти суму та
    // кількість ненульових елементів
 for (int i = 1; i <= N; i++) //цикл для обробки елементів послідовності:
{
     cout << "a" << i << " = ";
     cin >> a;            //введення поточного елементу
     if (a != 0)          //обробка поточного елементу
     {
          Avg += a; // формування суми ненульових елементів
          k++;      // та їх кількості
     }
}
 if (k == 0)   //опрацювання результату – перевірка, чи існує середнє арифметичне
ненульових
     cout << "All elements are 0" << endl;
else
     cout << "Avg =" << Avg / k << endl;
```

Мінімум в послідовності довжиною N (використовується метод пошуку мінімального/максимального елементу)

```
cout << "a1 = "; // підготовка початкового значення для розрахунку мінімуму:
cin >> a;    // перший елемент послідовності обробляється окремо
min = a;

for (int i = 2; i <= N; i++) //цикл для обробки елементів послідовності,
                    //перший елемент не обробляється повторно, починаємо з другого
{
     cout << "a"<<i<<" = ";
     cin >> a;              //введення поточного елементу
     if (a < min)           //обробка поточного елементу – перевірка елементу та
          min = a;    //зміна значення мінімуму
}

cout << "Min =" << min <<endl; //опрацювання результату – виведення мінімуму
```

Максимум в послідовності довжиною N (використовується метод пошуку мінімального/максимального елементу)

```
cout << "a1 = "; // підготовка початкового значення для розрахунку максимуму:
cin >> a;    // перший елемент послідовності обробляється окремо
max = a;

for (int i = 2; i <= N; i++) //цикл для обробки елементів послідовності:
                    //перший елемент не обробляється повторно, починаємо з другого
{
     cout << "a"<<i<<" = ";
     cin >> a;              //введення поточного елементу
     if (a > max)           //обробка поточного елементу – перевірка елементу та
          max = a;     //зміна значення максимуму
}

cout << "Max =" << max <<endl; //опрацювання результату – виведення максимуму
```