**Реалізація рекурсивних програм**

В основі реалізації рекурсивної підпрограми лежить структура даних, що називається **стеком,** в якому зберігаються всі не глобальні дані, що беруть участь у всіх викликах підпрограми, при яких вона ще не завершила свою роботу.

Кожен виклик підпрограми використовує фрагмент стека, довжина якого залежить від підпрограми, що викликається. У **загальному випадку** при виклику процедурою А процедури В відбувається наступне:

1. В вершину стека поміщається фрагмент потрібного розміру. До нього входять такі, дані:

(а) показники фактичних параметрів виклику процедури В;

(б) порожні комірки для локальних змінних, визначених у процедурі В;

(в) адреса повернення (АВ), тобто адреса команди в процедурі А, яку слід виконати після того, як процедура В закінчить свою роботу.

Якщо В – функція, то у фрагмент стеку для В поміщається покажчик осередку у фрагменті стека для А, в якій належить помістити значення цієї функції (адреса значення).

2. Управління передається першому оператору процедури В.

3. При завершенні роботи процедури В управління передається

процедурі А за допомогою наступної послідовності кроків:

(а) адреса повернення витягується з вершини стеку;

(б) якщо В – функція, то її значення запам'ятовується в комірці, на яку вказує покажчик адреси значення;

(в) фрагмент стека процедури В витягується з стека, в вершину ставиться фрагмент процедури А;

(г) виконання процедури А поновлюється з команди, зазначеної в адресі повернення.

**При виклику підпрограмою самої себе,** тобто в рекурсивному випадку,

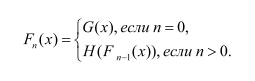
виконується та ж сама послідовність дій.

**Рекурсивне визначення** складається з двох незалежних частин: базової і

рекурсивної.

Базова частина не є рекурсивним ствердженням, вона задає визначення для деякої фіксованої групи об'єктів і визначає умову закінчення рекурсії. У першому прикладі в базовій частині стверджується, що об'єкт 0! = 1.

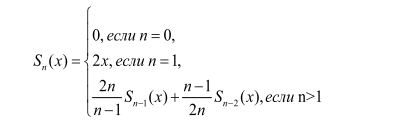
**Рекурсія – ітерація**

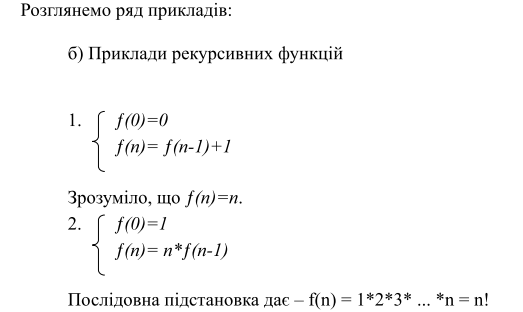
Клас функцій, що мають визначення виду

може завжди бути виражений ітеративно так, що застосування рекурсії необов'язково.

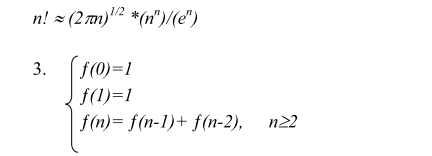
Приклад. Необхідно скласти визначення функції для обчислення значень

деяких поліномів, визначення яких має наступний вигляд:





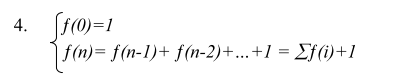
формула Стірлінга для наближеного обчислення факторіала для великих n:

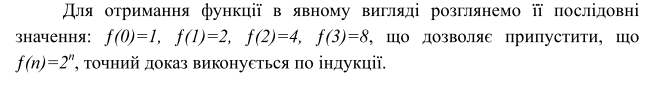


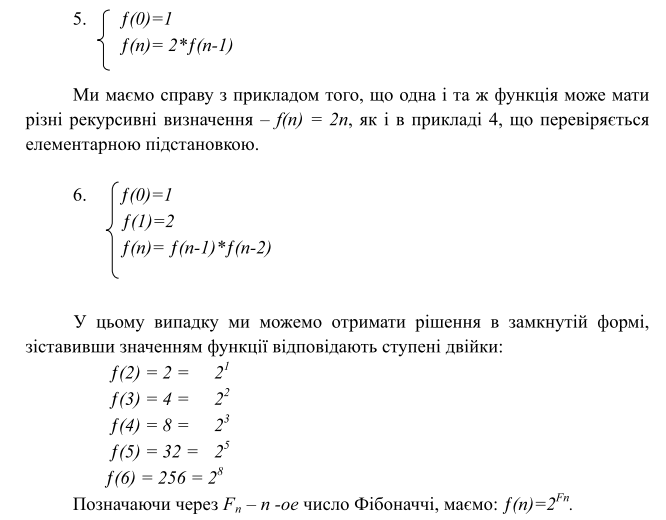
Ця рекурсивна функція визначає числа Фібоначчі: *1 1 2 3 5 8 13*, які

досить часто виникають при аналізі різних завдань, у тому числі і при аналізі

алгоритмів. Відзначимо, що асимптотично *f (n)* » [1,618 n]







**Аналіз трудоміскості рекурсивних алгоритмів.**

Один з методів аналізу трудомісткості рекурсивного алгоритму, який будується на основі підрахунку вершин рекурсивного дерева.

Для оцінки трудомісткості рекурсивних алгоритмів будується повне дерево рекурсії. Воно являє собою граф, вершинами якого є набори фактичних параметрів при всіх викликах функції, починаючи з першого звернення до неї, а ребрами – пари таких наборів, відповідних взаємним викликам. При цьому вершини дерева рекурсії відповідають фактичним викликам рекурсивних функцій.

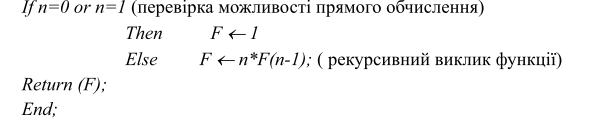
Воно являє собою граф, вершинами якого є набори фактичних параметрів при всіх викликах функції, починаючи з першого звернення до неї, а ребрами – пари таких наборів, відповідних взаємним викликам. При цьому вершини дерева рекурсії відповідають фактичним викликам рекурсивних функцій. Слід зауважити, що одні й ті ж набори параметрів можуть відповідати різним вершинам дерева. Корінь повного дерева рекурсивних викликів – це вершина повного дерева рекурсії, відповідна початковим зверненням до функції.

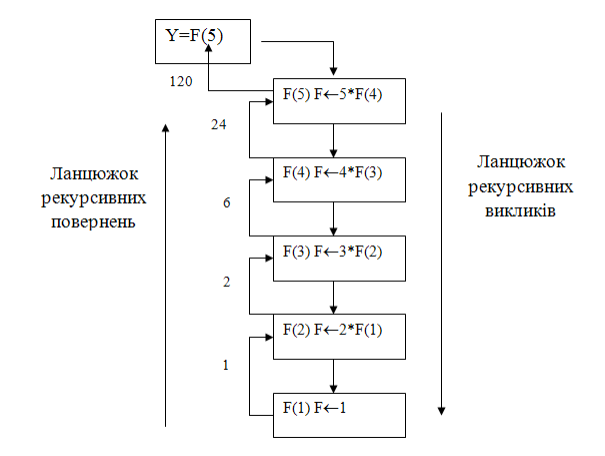
Важливою характеристикою рекурсивного алгоритму є глибина рекурсивних викликів – найбільше одночасне кількість рекурсивних звернень функції, що визначає максимальну кількість шарів рекурсивного стека, в якому здійснюється зберігання відкладених обчислень. Кількість елементів повних рекурсивних звернень завжди не менше глибини рекурсивних викликів. При розробці рекурсивних програм необхідно враховувати, що глибина рекурсивних викликів не повинна перевершувати максимального розміру стека використовуваного обчислювального середовища.

При цьому обсяг рекурсії – це одна з характеристик складності рекурсивних обчислень для конкретного набору параметрів, що представляє собою кількість вершин повного рекурсивного дерева без одиниці.

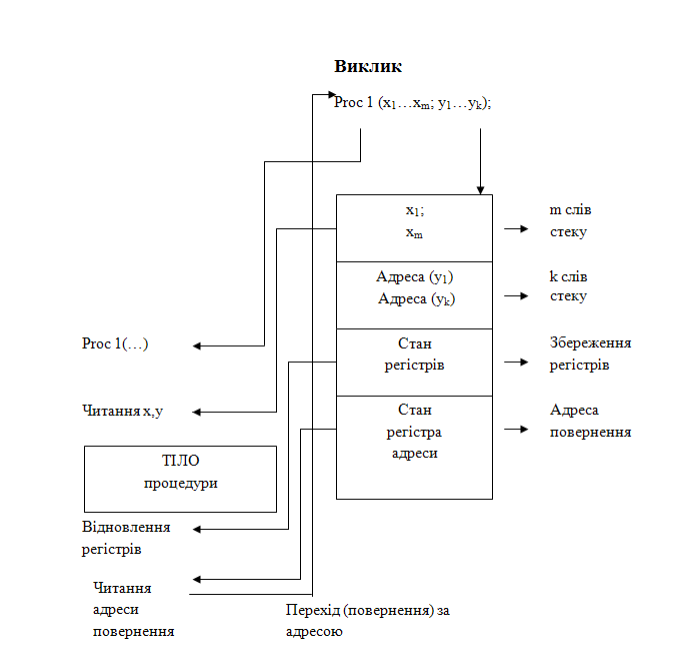
Рекурсивний алгоритм може бути реалізований й ітераційною послідовністю дій.

Розглянемо приклад рекурсивної функції, що обчислює факторіал: F(n).

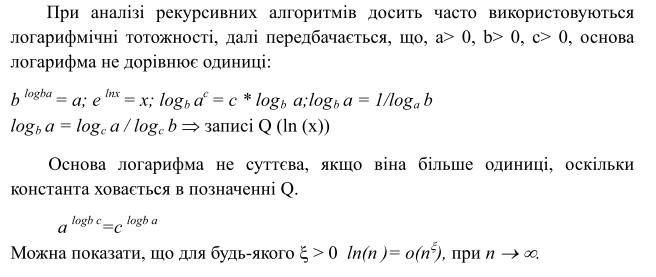




Дерево рекурсії при обчисленні факторіалу



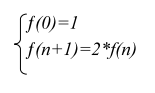
Виклик процедури з використанням програмного стека

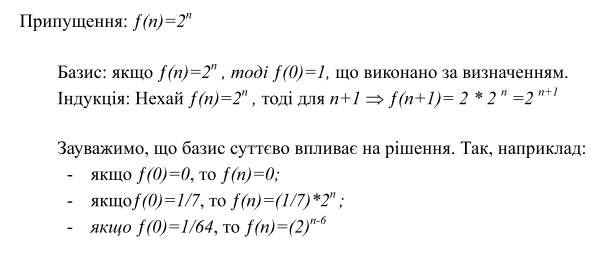


Метод індукції

Метод полягає в тому, що б спочатку вгадати рішення, а потім довести його правильність за допомогою індукції.

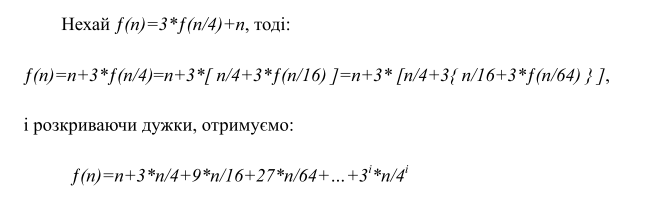
Приклад:

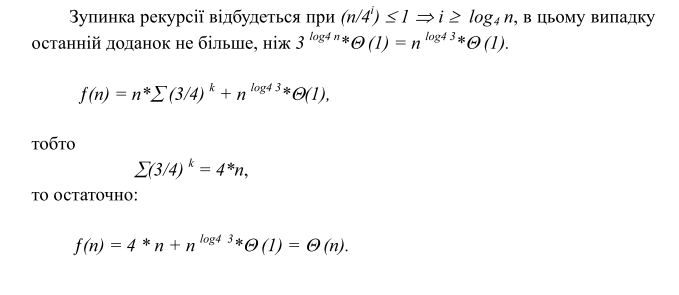


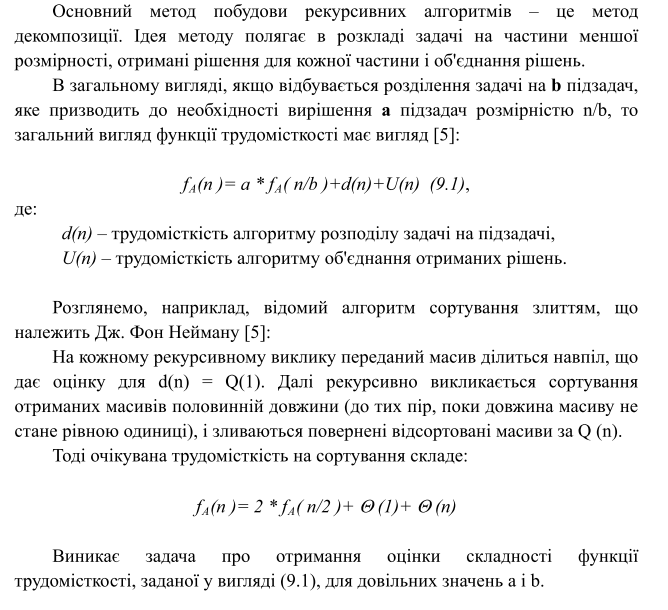


Метод ітерації

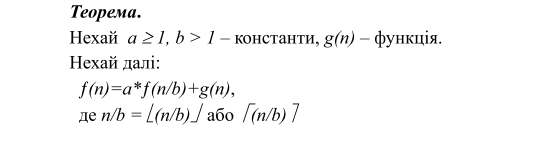
Суть методу полягає в послідовній підстановці – ітерації рекурсивного визначення, з наступним виявленням загальних закономірностей:

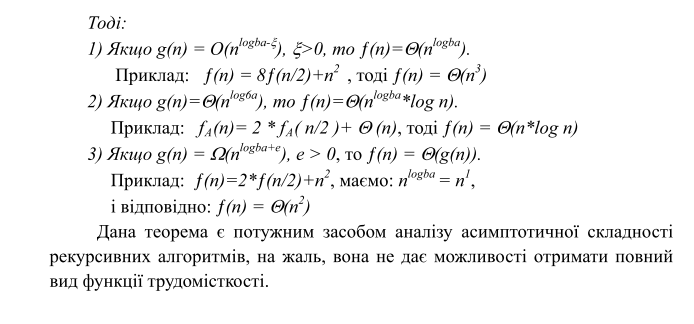


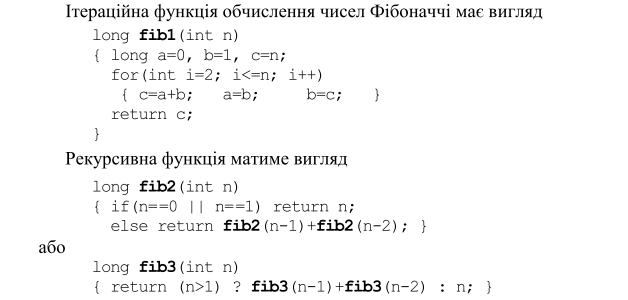




Теорема про рекурентні співвідношення

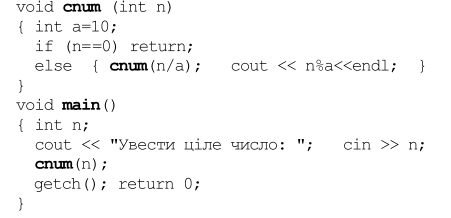




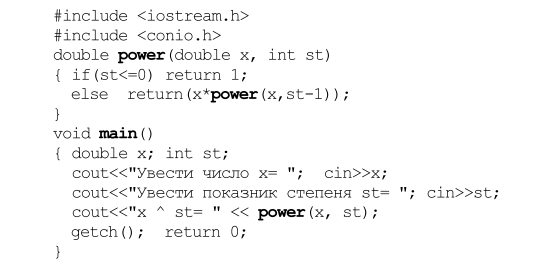


Приклади рекурсії

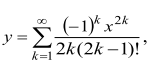
Увести ціле число і вивести всі цифри цього числа

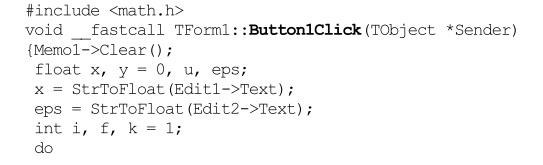
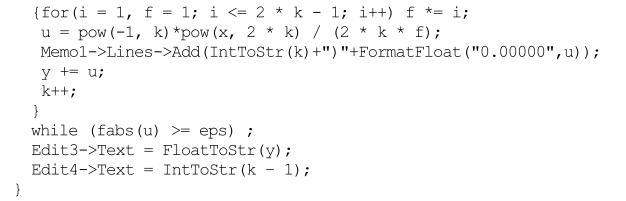


Функція піднесення числа до степеня



Використання рекурентних множників

Приклад. Обчислити суму ряду  підсумовуючи члени ряду, значення яких за модулем є більшими за задану точність ε . Визначити кількість доданків. Значення х (– 2 < x < 2) та ε = 10-4

 Визначення рекурентного множника

