
Алгоритми та структури даних. Основи алгоритмізації

Додаток 1

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний
інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 6 з дисципліни
«Алгоритми та структури даних-1.
Основи алгоритмізації»

«Дослідження лінійних алгоритмів»

Варіант 9

Виконав студент Григоренко Родіон Ярославович
(шифр, прізвище, ім'я, по батькові)

Перевірів Вечерковська А.С.
(прізвище, ім'я, по батькові)

Київ 2021

Алгоритми та структури даних. Основи алгоритмізації

Лабораторна робота 6

Дослідження рекурсивних алгоритмів

Мета - дослідити особливості роботи рекурсивних алгоритмів та набути практичних навичок їх використання під час складання програмних специфікацій підпрограм.

Індивідуальне завдання

9. Обчислити добуток 7 елементів арифметичної прогресії, що зростає: початкове значення – 1, крок – 3

Варіант 9 Постановка задачі

Результатом є виведення добутку 7 перших членів арифметичної прогресії, знайденого за допомогою рекурсивної функції.

Побудова математичної моделі

Складемо таблицю імен змінних та функцій

Змінна	Тип	Ім'я	Призначення
Перший член прогресії	Натуральний	firstNum	Дане
Крок прогресії	Натуральний	Step	Дане
Задана глибина розрахунку добутку членів прогресії	Натуральний	Depth	Дане
Поточна глибина розрахунку добутку членів прогресії	Натуральний	Dep	Проміжні дані
Добуток членів прогресії	Натуральний	Dob	Результат
Рекурсивна функція для знаходження добутку членів прогресії	Натуральний	Prog	Розрахунок добутку членів прогресії
Перший параметр функції Prog	Натуральний	num	Проміжні дані
Другий параметр функції Prog	Натуральний	step	Проміжні дані

Алгоритми та структури даних. Основи алгоритмізації

Третій параметр функції Prog	Натуральний	depth	Проміжні дані
Четвертий параметр функції Prog	Натуральний	dob	Проміжні дані
П'ятий параметр функції Prog	Натуральний	dep	Проміжні дані

num,step,depth,dob,dep - параметри рекурсивної функції Prog

Їм відповідають змінні firstNum, Step, Depth ,Dob ,Dep

Математичне формулювання зводиться до рекурсивного обчислення кожного наступного члена прогресії та добутку знайдених членів,поки глибина рекурсії не більша за задану.

Розв'язання

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо функцію Prog

Крок 3. Деталізуємо рекурсію

Псевдокод

Крок 1

функція Prog(num, step, depth, dob ,dep)

Обчислення добутку членів прогресії

все функція

початок

firstNum = 1, Step = 3, Depth = 7, Dob = 1,Dep = 1

Dob = Prog(firstNum, Step, Depth, Dob, Dep)

виведення Dob

кінець

Крок 2

функція Prog(num, step, depth, dob ,dep)

якщо dep <= depth

то

Рекурсивний виклик функції Prog

інакше

повернути dob

все якщо

все функція

Алгоритми та структури даних. Основи алгоритмізації

початок

firstNum = 1, Step = 3, Depth = 7, Dob = 1, Dep = 1

Dob = Prog(firstNum, Step, Depth, Dob, Dep)

виведення Dob

кінець

Крок 3

функція Prog(num, step, depth, dob ,dep)

якщо dep <= depth

то

Prog(num + step, step, depth, dob*num,dep + 1)

інакше

повернути dob

все якщо

все функція

початок

firstNum = 1, Step = 3, Depth = 7, Dob = 1, Dep = 1

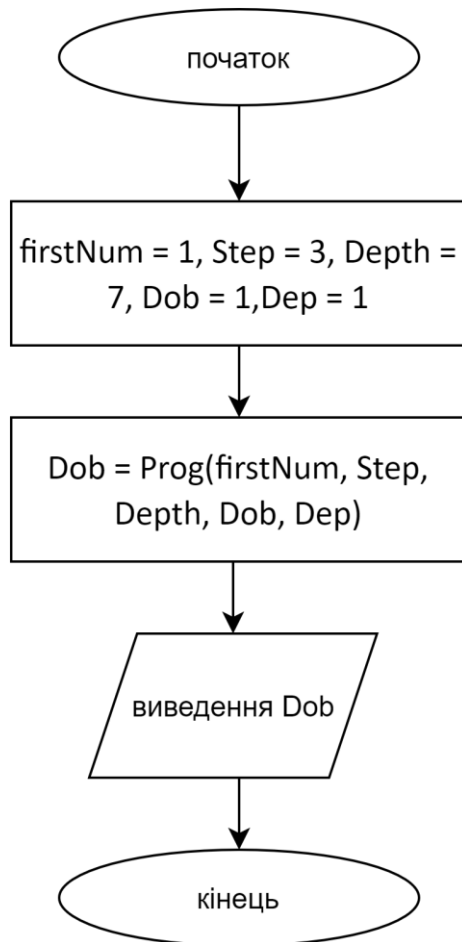
Dob = Prog(firstNum, Step, Depth, Dob, Dep)

виведення Dob

кінець

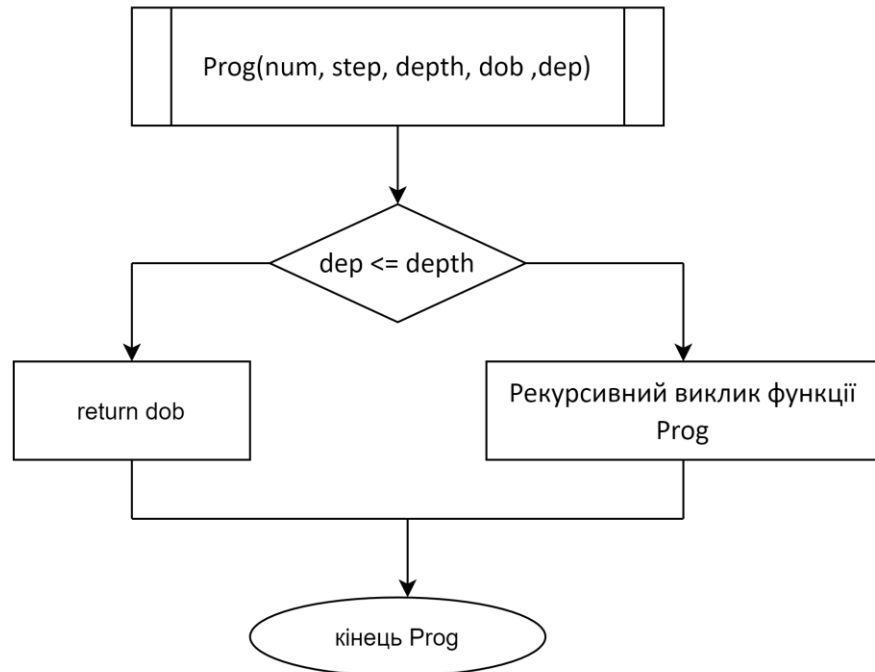
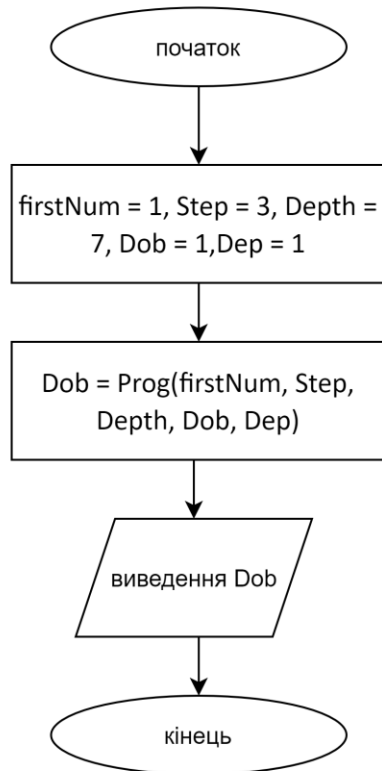
Блок-схема

Крок 1

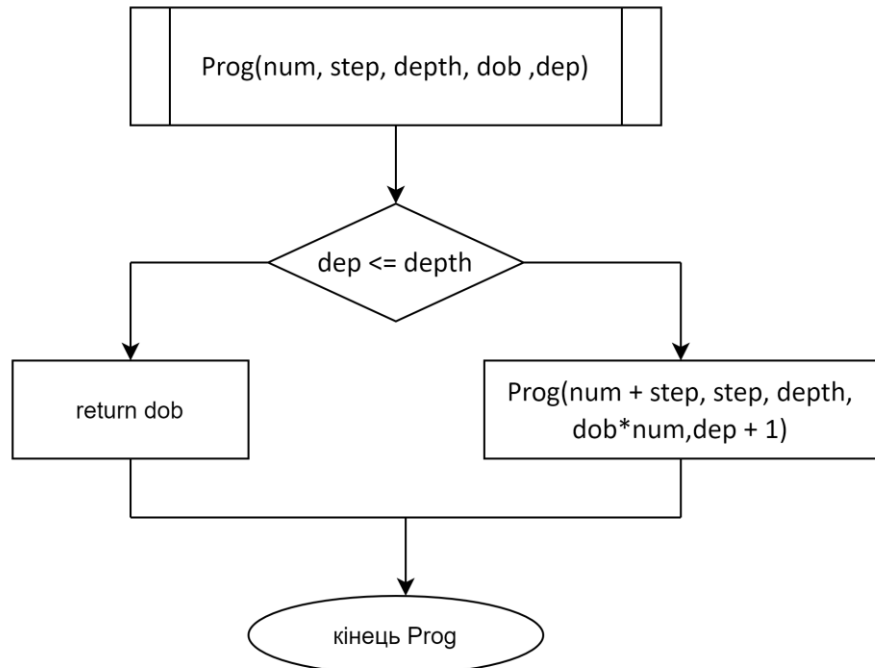
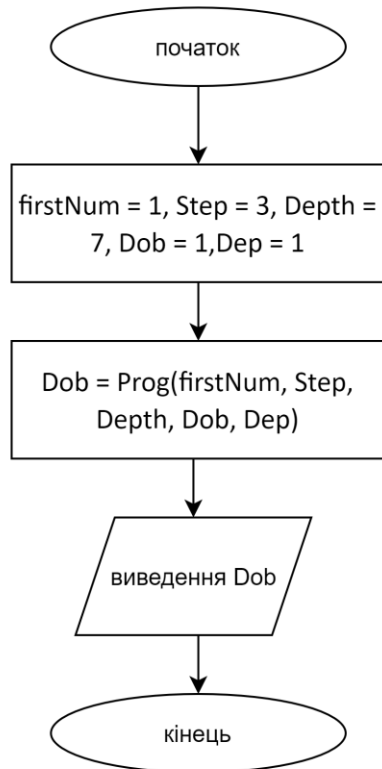


Алгоритми та структури даних. Основи алгоритмізації

Крок 2



Крок 3



Алгоритми та структури даних. Основи алгоритмізації

Тестування

Блок	Дія
	Початок
	firstNum = 1, Step = 3, Depth = 7, Dob = 1, Dep = 1
Глибина рекурсії	Функція Prog
1	num = 1 step = 3 depth = 7 dob = 1 dep = 1
2	num = 4 step = 3 depth = 7 dob = 1 dep = 2
3	num = 7 step = 3 depth = 7 dob = 4 dep = 3
4	num = 10 step = 3 depth = 7 dob = 28 dep = 4
5	num = 13 step = 3 depth = 7 dob = 280 dep = 5
6	num = 16 step = 3 depth = 7 dob = 3640 dep = 6

Алгоритми та структури даних. Основи алгоритмізації

7	num = 19 step = 3 depth = 7 dob = 58240 dep = 7
8	num = 21 step = 3 depth = 7 dob = 1106560 dep = 8
	8 > 7, отже, функція повертає значення dob = 1106560
	Виведення: 1106560
	Кінець

Код

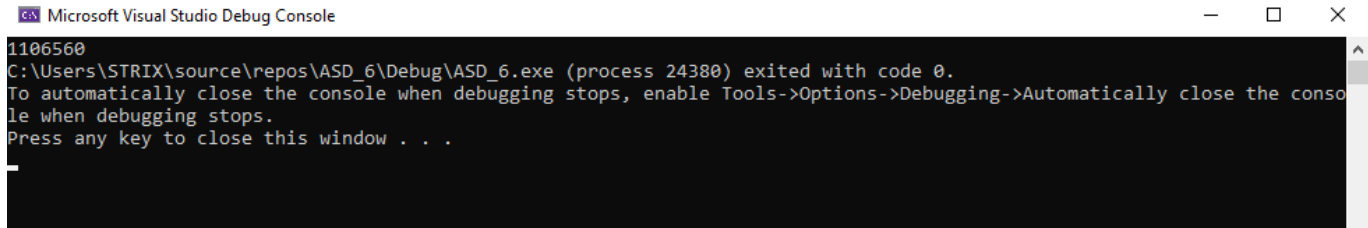
```
#include <iostream>
using namespace std;

int Prog(int num, int step, int depth, int dob ,int dep) {
    if (dep <= depth) {
        return Prog(num + step, step, depth, dob*num, dep + 1);
    }
    else {
        return dob;
    }
}

int main()
{
    int firstNum = 1, Step = 3, Depth = 7, Dob = 1, Dep = 1;
    Dob = Prog(firstNum, Step, Depth, Dob, Dep);
    cout << Dob;
}
```

Алгоритми та структури даних. Основи алгоритмізації

Результат коду



```
Microsoft Visual Studio Debug Console
1106560
C:\Users\STRIX\source\repos\ASD_6\Debug\ASD_6.exe (process 24380) 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 . . .
```

Висновки

Я дослідив особливості роботи рекурсивних алгоритмів та набув практичних навичок їх використання під час складання програмних специфікацій підпрограм. Побудував мат. модель, псевдокод та блок схему. Протестував алгоритм.