

Алгоритми та структури даних. Основи алгоритмізації

Міністерство освіти і науки України
Національний технічний університет України «Київський
політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації
і управління

Звіт

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

«Дослідження рекурсивних алгоритмів»

Варіант __4__

Виконав студент _____ Берлінський Ярослав Владленович _____
(шифр, прізвище, ім'я, по батькові)

Перевірив _____
(прізвище, ім'я, по батькові)

Київ 2020

Назва роботи: дослідження рекурсивних алгоритмів.

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

Варіант: 4

Умова задачі:

4. Обчислити спільний дільник для двох цілих десяткових чисел

Постановка задачі.

З клавіатури вводяться два цілих числа a і b . Треба обчислити їх найбільший спільний дільник(НСД) та вивести.

Побудова математичної моделі: для більшої наочності складемо таблицю імен змінних.

Змінна	Тип	Ім'я	Призначення
Перше число	Цілий	a	Початкові дані
Друге число	Цілий	b	Початкові дані

Фактично задана умова зводиться до послідовного знаходження найбільшого спільного дільника. Інтерпретуємо рішення задачі математичною моделлю:

Найбільший спільний дільник(НСД) двох чисел це найбільше число, що ділить обидва дані числа без остачі.

НСД знаходиться різними способами: від перебору чисел для двох заданих та їх порівняння до алгоритму Евкліда). Скористаємося алгоритмом Евкліда.

Математична інтерпретація роботи алгоритму:

- задано **два цілих числа a і b** . Алгоритм Евкліда заснований на тому, що НСД не змінюється, якщо знайти остачу від ділення більшого числа на менше. Оскільки більше з двох чисел постійно зменшується, повторне виконання цього кроку дає все менші числа, поки одне з них не дорівнюватиме нулю. Коли одне з чисел дорівнюватиме нулю, те, що залишилось, і є НСД:

НСД(a ; b) :

$$\text{якщо } a > b \rightarrow a = a \bmod b \quad (1)$$

$$\text{якщо } b < a \rightarrow b = b \bmod a \quad (2)$$

$$\text{якщо } \begin{cases} a = 0 \\ b = 0 \end{cases} \rightarrow \text{НСД}(a; b) = a + b \quad (3)$$

Важливо підмітити, що $\text{НСД}(a, b) = 1$, тоді a та b називають взаємно простими. Ця властивість не залежить від того, чи прості числа a та b .

Основний етап алгоритму – знаходження НСД – буде реалізований **рекурентно за допомогою підпрограм**, де умови(1) і (2) будуть рекурентно викликати функцію(саме функцію, а не процедуру, адже буде повернене одне значення), а умова(3) стане **базовим(термінальним) випадком**, що свідчить про скінченність заданого алгоритму.

Отже, проаналізувавши основні етапи алгоритму в математичній моделі, перейдемо до роз'язання задачі.

Розв'язання.

1. *Програмні специфікації запишемо у псевдокодi та у графічній формi у вигляді блок-схеми.*

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо ввід чисел.

Крок 3. Деталізуємо дію знаходження НСД двох чисел у підпрограмі.

Псевдокод

Крок №1

Підпрограма:

F(a,b):

початок

знаходження НСД(a;b)

кінець

Програма:

початок

введення a і b

перевірка на цілочисельність та вивід

кінець

Крок №2

Підпрограма:

F(a,b):

початок

знаходження НСД(a;b)

кінець

Програма:

початок

введення a і b

якщо $a < 0 \parallel b < 0$

a:=fabs(a)

b:=fabs(b)

вивести F(a,b)

інакше

вивести F(a,b)

все якщо

кінець

Крок №3

Підпрограма:

F(a,b):

початок

якщо $a*b=0$

повернути «НСД = », $a+b$

все якщо

якщо $a < b$

повернути $F(a, b \bmod a)$

інакше

повернути $F(a \bmod b, b)$

все якщо

кінець

Програма:

початок

введення a і b

якщо $a < 0 \parallel b < 0$

$a := \text{fabs}(a)$

$b := \text{fabs}(b)$

вивести $F(a,b)$

інакше

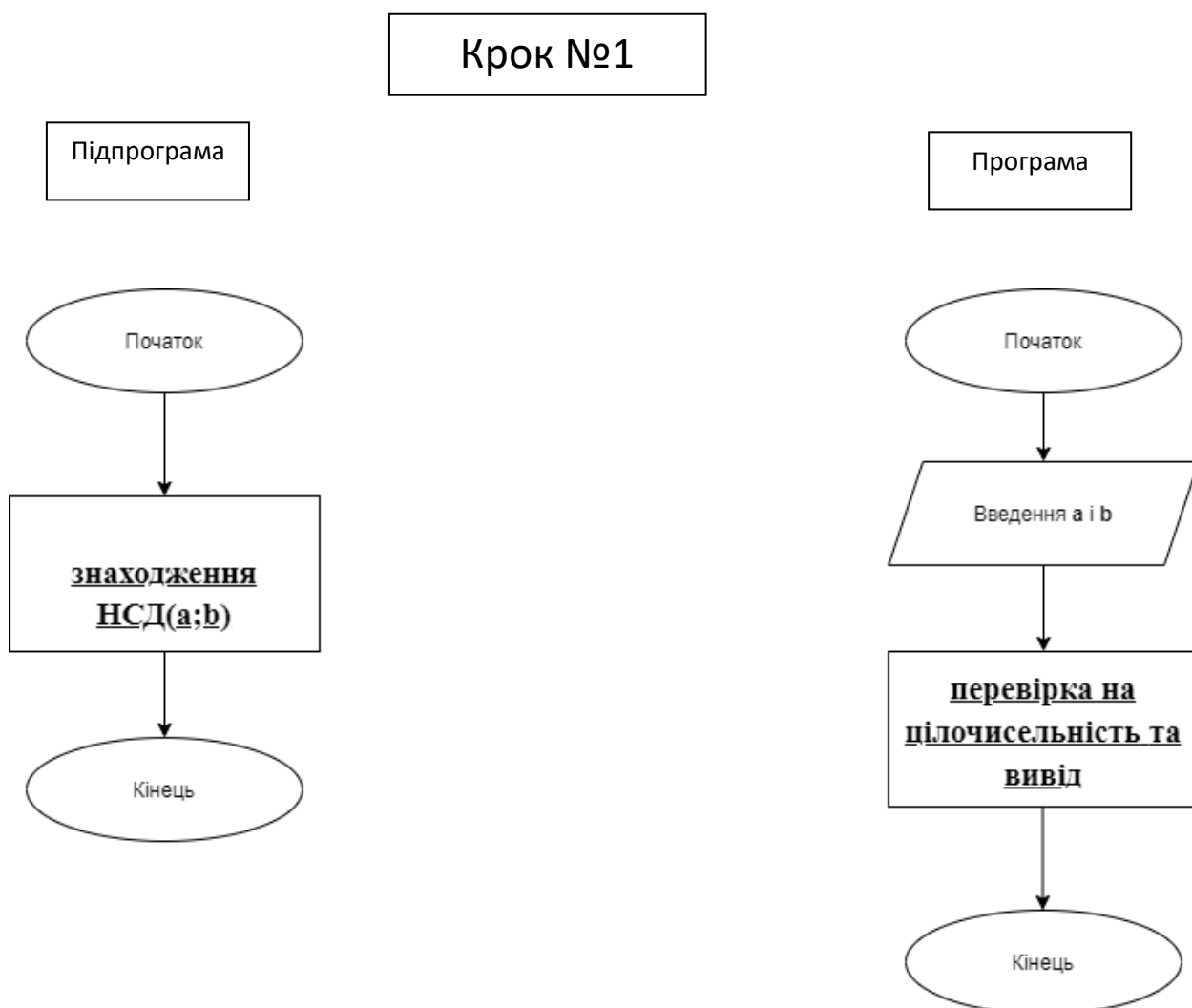
вивести $F(a,b)$

все якщо

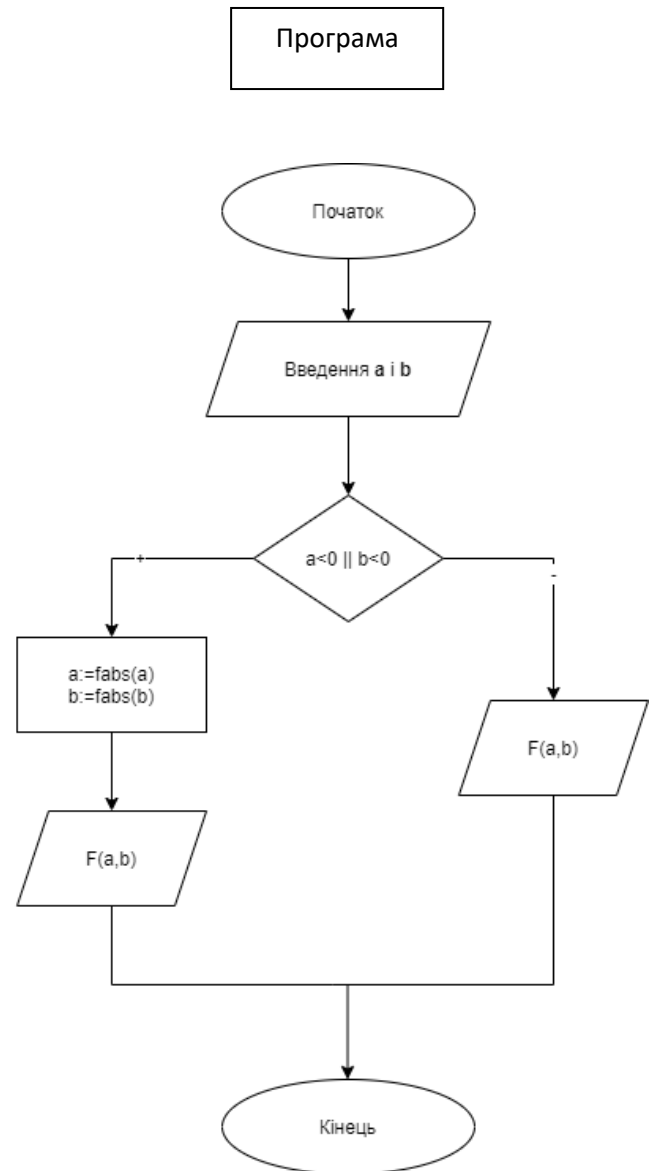
кінець

Утворивши псевдокод, побудуємо блок-схему алгоритму.

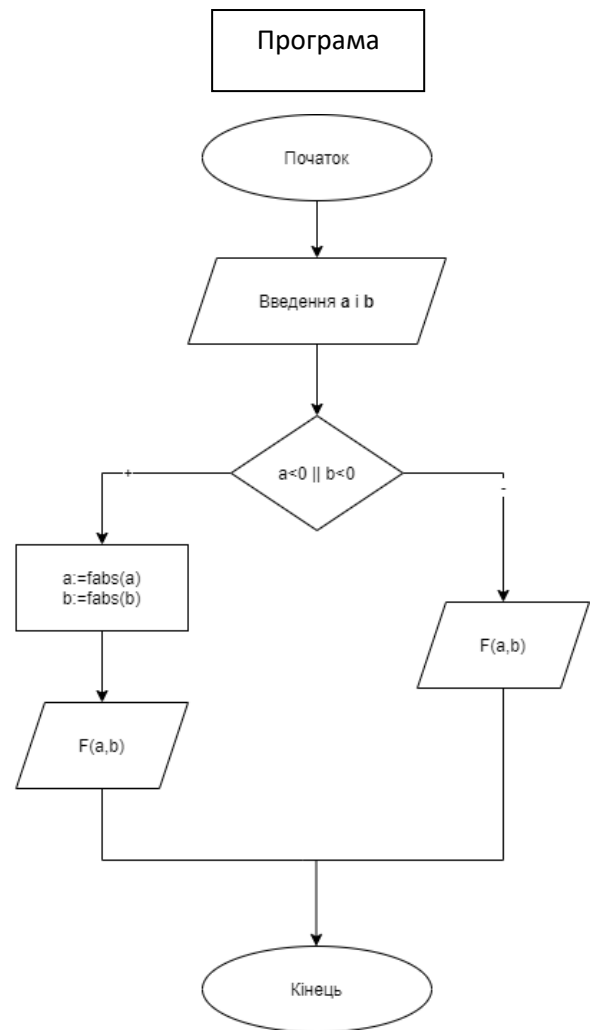
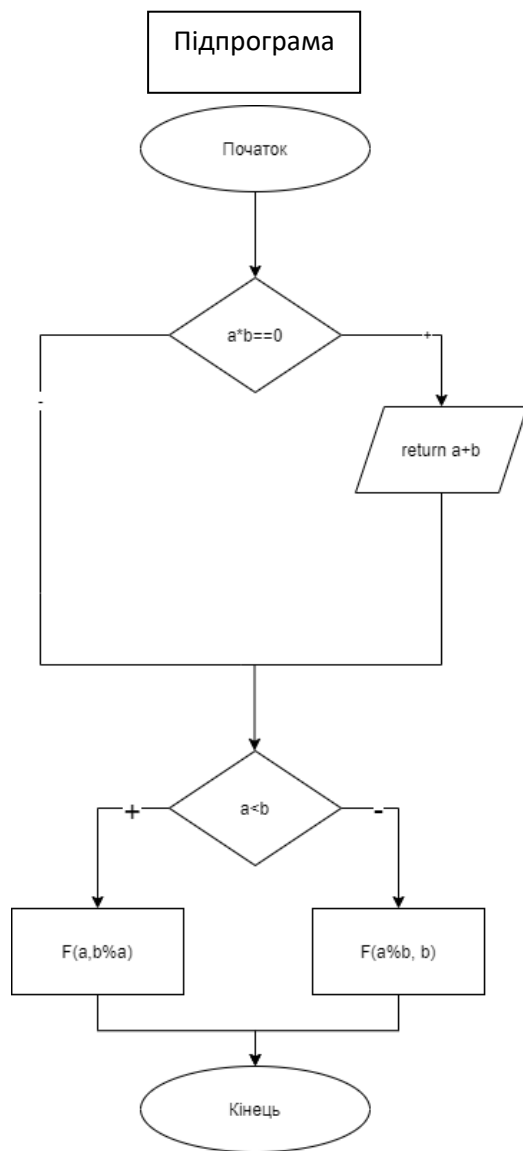
Блок-схема



Крок №2



Крок №3



Випробування алгоритму

Перевіримо правильність алгоритму на довільних конкретних значеннях початкових даних.

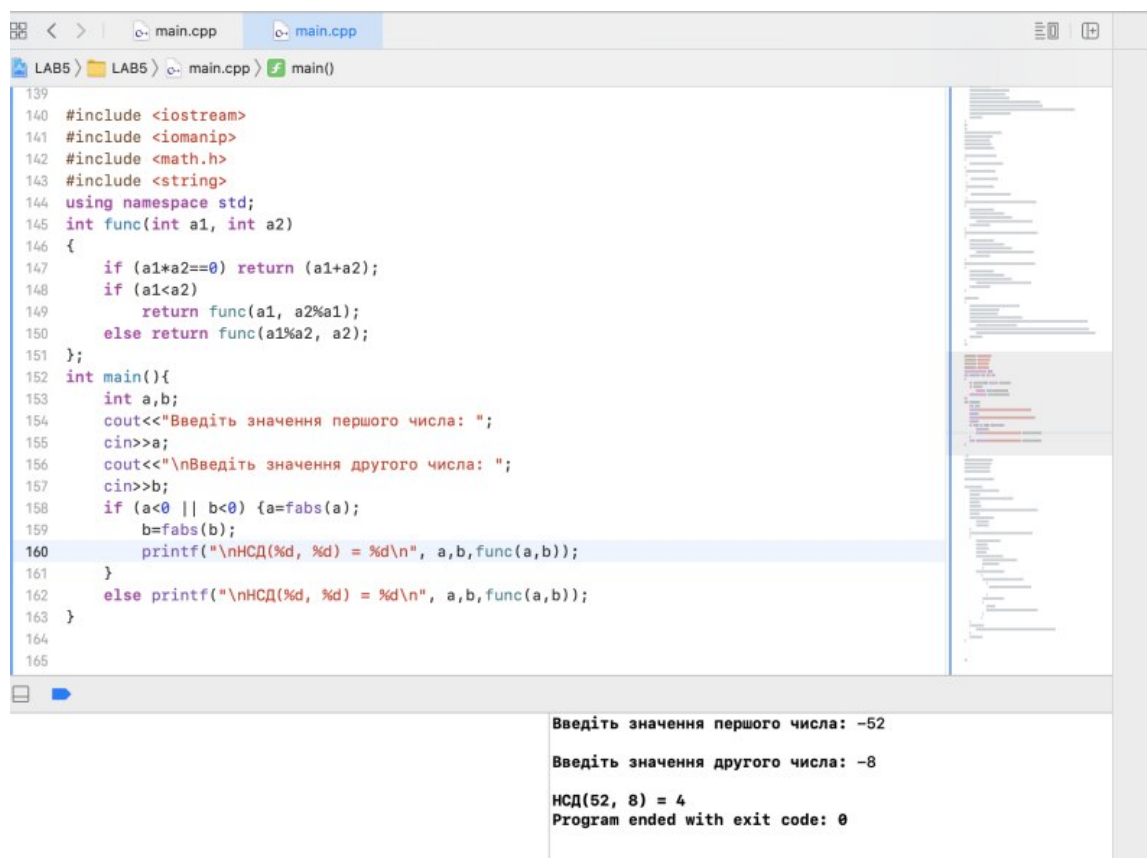
Тест №1($a=-52;n=-8$)

Блок	Дія
1	Початок
2	Введення $a=-52, n=-8$
3	$a < 0 \ \ b < 0 \rightarrow \text{true} \rightarrow a:=52, b:=8 \rightarrow F(52,8)$
4	Ініціалізація підпрограми $F(52,8)$
5	$52 * 8 == 0 \rightarrow \text{false}$
6	$52 > 8 \rightarrow \text{true} \rightarrow F(52 \% 8, 8)$
7	Ініціалізація підпрограми $F(52 \% 8, 8)$
8	$4 * 8 \rightarrow \text{false}$
9	$4 > 8 \rightarrow \text{false} \rightarrow F(4, 8 \% 4)$
10	Ініціалізація підпрограми $F(4, 8 \% 4)$
11	$4 * 0 = 0 \rightarrow \text{return } 4$
12	Вивід «НСД = », 4
13	Кінець

Код програми:

```
140 #include <iostream>
141 #include <iomanip>
142 #include <math.h>
143 #include <string>
144 using namespace std;
145 int func(int a1, int a2)
146 {
147     if (a1*a2==0) return (a1+a2);
148     if (a1<a2)
149         return func(a1, a2%a1);
150     else return func(a1%a2, a2);
151 };
152 int main(){
153     int a,b;
154     cout<<"Введіть значення першого числа: ";
155     cin>>a;
156     cout<<"\nВведіть значення другого числа: ";
157     cin>>b;
158     if (a<0 || b<0) {a=fabs(a);
159         b=fabs(b);
160     printf("\nНСД(%d, %d) = %d\n", a,b,func(a,b));
161     }
162     else printf("\nНСД(%d, %d) = %d\n", a,b,func(a,b));
163 }
164
165
```

Тест№1(a=-52, b=-8):



```
139
140 #include <iostream>
141 #include <iomanip>
142 #include <math.h>
143 #include <string>
144 using namespace std;
145 int func(int a1, int a2)
146 {
147     if (a1*a2==0) return (a1+a2);
148     if (a1<a2)
149         return func(a1, a2%a1);
150     else return func(a1%a2, a2);
151 };
152 int main(){
153     int a,b;
154     cout<<"Введіть значення першого числа: ";
155     cin>>a;
156     cout<<"\nВведіть значення другого числа: ";
157     cin>>b;
158     if (a<0 || b<0) {a=fabs(a);
159         b=fabs(b);
160     printf("\nНСД(%d, %d) = %d\n", a,b,func(a,b));
161     }
162     else printf("\nНСД(%d, %d) = %d\n", a,b,func(a,b));
163 }
164
165
```

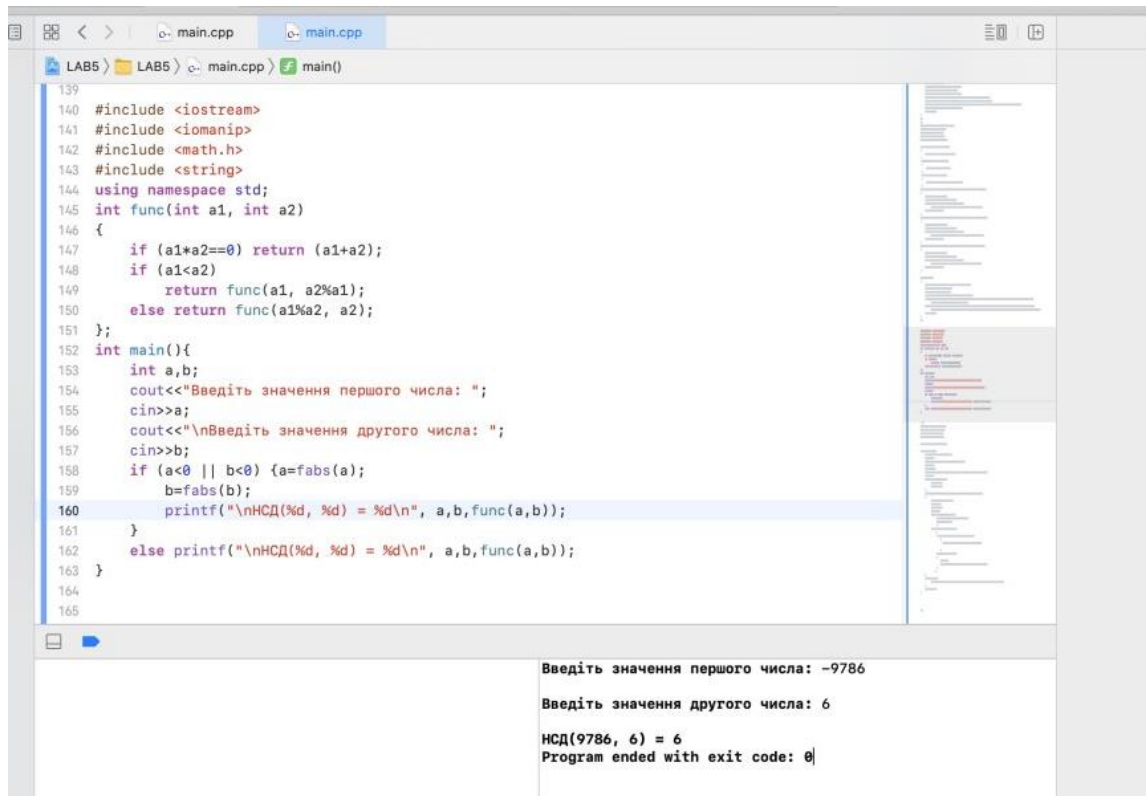
Введіть значення першого числа: -52

Введіть значення другого числа: -8

НСД(52, 8) = 4

Program ended with exit code: 0

Тест№2(a=-9786, b=6):



```
139
140 #include <iostream>
141 #include <iomanip>
142 #include <math.h>
143 #include <string>
144 using namespace std;
145 int func(int a1, int a2)
146 {
147     if (a1*a2==0) return (a1+a2);
148     if (a1<a2)
149         return func(a1, a2%a1);
150     else return func(a1%a2, a2);
151 };
152 int main(){
153     int a,b;
154     cout<<"Введіть значення першого числа: ";
155     cin>>a;
156     cout<<"\nВведіть значення другого числа: ";
157     cin>>b;
158     if (a<0 || b<0) {a=fabs(a);
159                     b=fabs(b);
160     printf("\nНСД(%d, %d) = %d\n", a,b,func(a,b));
161     }
162     else printf("\nНСД(%d, %d) = %d\n", a,b,func(a,b));
163 }
164
165
```

Введіть значення першого числа: -9786
Введіть значення другого числа: 6
НСД(9786, 6) = 6
Program ended with exit code: 0

Інші тести можна виконати самостійно, перейшовши за посиланням до коду на [GitHub](#)

Висновок: отже, за допомогою підпрограми(рекурсивна функція) було організовано знаходження НСД для двох заданих цілих чисел a і b .

Аналізуючи рекурсивну функцію, приходимо до висновку, що вона значно спрощує написання алгоритму підпрограми обчислення(і не тільки НСД), але є більш затратною на пам'ять комп'ютера та кількість операцій. Ітераційна форма при своїй громіздкості стає більш швидкою, адже не звертається до стеку і не створює додаткового навантаження на внутрішню пам'ять комп'ютера.

Окрім цього, було зроблено автоматичне виправлення введених від'ємних чисел(щоб не призвести програму до нескінченного повторення дій функції).