НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА**

з дисципліни «СИСТЕМНЕ ПРОГРАМУВАННЯ-2».

на тему: «Розробка компілятора підмножини команд мови Python, що містять арифметичні дії і дужкові форми»

Студента 3 курсу групи ІВ-93

спеціальності

123 «Комп’ютерна  інженерія»

Калайда Т.В.

Керівник      доцент Павлов В.Г.

Національна оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів \_\_\_\_\_\_\_\_\_\_

Оцінка:  ECTS   \_\_\_\_\_\_\_\_\_\_\_

                                                                        Київ- 2021 рік

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет (інститут) інформатики та обчислювальної техніки

( повна назва )

Кафедра обчислювальної техніки

( повна назва )

Освітньо-кваліфікаційний рівень бакалавр

Спеціальність 123 «Комп’ютерна інженерія»

*(шифр і назва)*

**ЗАВДАННЯ**

НА КУРСОВУ РОБОТУ СТУДЕНТУ

Калайди Тараса Вікторовича

*(прізвище, ім’я, по батькові)*

1. Тема роботи «Розробка компілятора підмножини команд мови Python, що містять арифметичні дії і дужкові форми»

керівник роботи Павлов В.Г. к.т.н.**,** доцент

( прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи ----- 2021 р.

3. Вхідні дані до роботи

- вхідна мова: Python;

- цільова мова: MASM32 Assembler x86;

- типи конструкцій: арифметичні дії, розгалуження, цикли, функції.

4. Зміст розрахунково-пояснювальної записки

- розробити лексичний аналізатор мови Python;

- розробити синтаксичний аналізатор мови Python;

- розробити генератор коду мови MASM32 Assembler x86.

5. Перелік графічного матеріалу;

- алгоритм лексичного аналізу;

- алгоритм синтаксичного аналізу;

6. Дата видачі завдання 15 жовтня 2021 р

**Зміст**

Завдання2

Зміст……………………..………………………………………………………………………………………………………..... 3

Вступ4

Опис роботи програми5

Приклад6

Лістинг програми11

Виконання завдання12

Перевірка роботи (НСД)15

Перевірка роботи (НСК)32

Висновок49

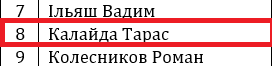
Джерела50

**Вступ**

В результаті виконання курсової роботи я маю отримати знання про теорію написання компіляторів та виконати поставлену задачу курсової роботи. Основний матеріла повинен взяти з матеріалів, наданих викладачем, тобто з файлу «Теорія побудови компіляторів».

Мною для виконання курсової роботи було створено компілятор мови Python на мові програмування Python. Дана програма читає вхідний файл з розширенням py, який містить код на мові Python. Далі програма виконує синтаксичний та лексичний аналіз вхідного тексту. Як кінцевий результат ми маємо код на masm32.

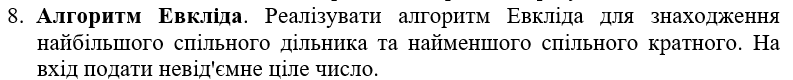
**Варіант №8 (місце у списку групи)**

****

**Завдання за варіантом №8**

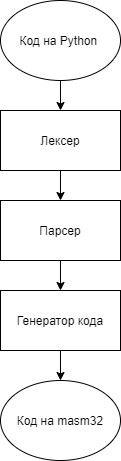


**Алгоритм за варіантом №8**



**Опис роботи програми**

Компілятор складається з лексера, парсера та генератора коду.



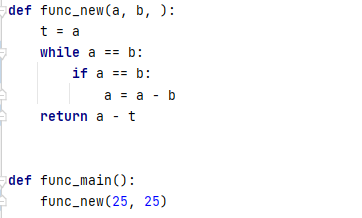
**Лексер** приймає вхідні данні на мові програмування python. Він працює з рядком, в якому розпізнає шаблони і діле на лексеми. В результаті утворюються токени.

**Парсер** отримує список токенів, які були утворені під час розпізнавання шаблонів лексером. Парсер створює базову структуру программи та повідомляє про наявні помилки.

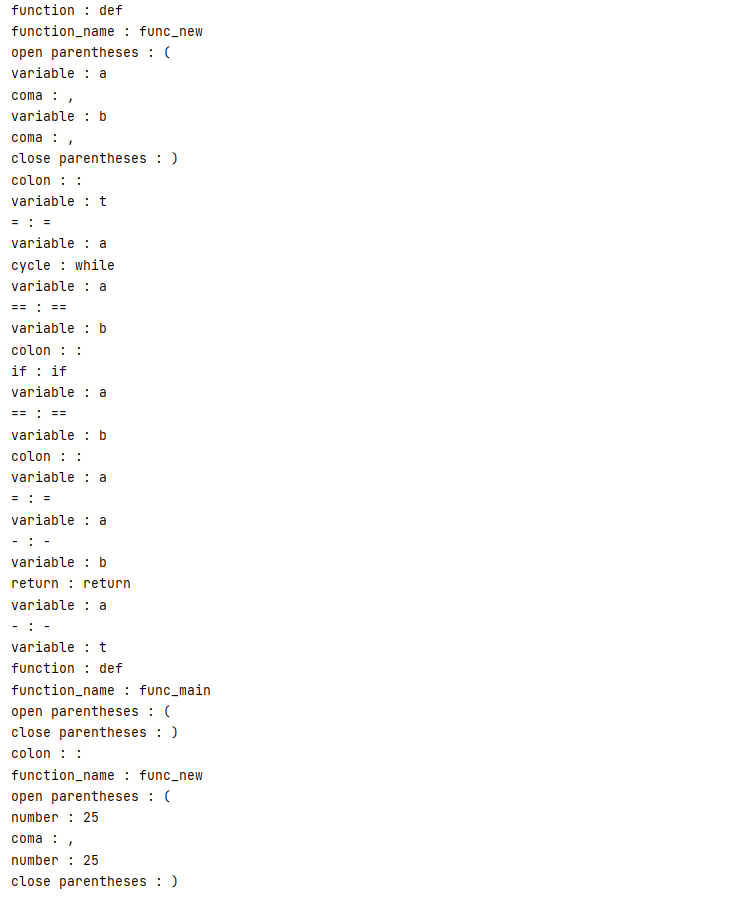
**Генератор коду** створює masm32 код.

**Приклад**

Код на мові Python



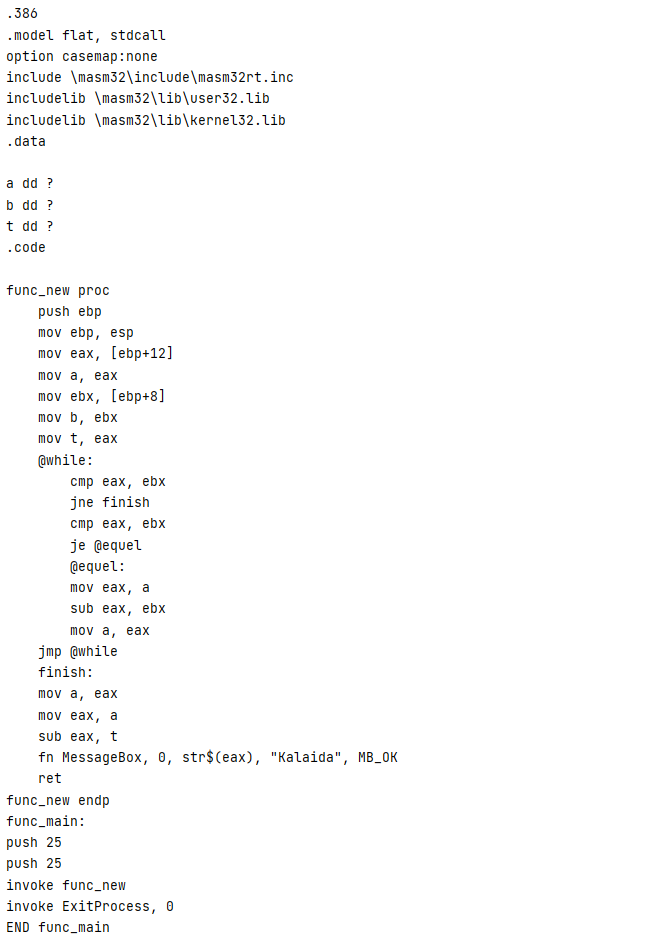
Отримані лексеми



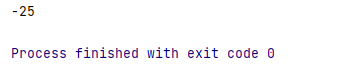
Синтаксичне дерево, яке створюється за допомогою парсеру

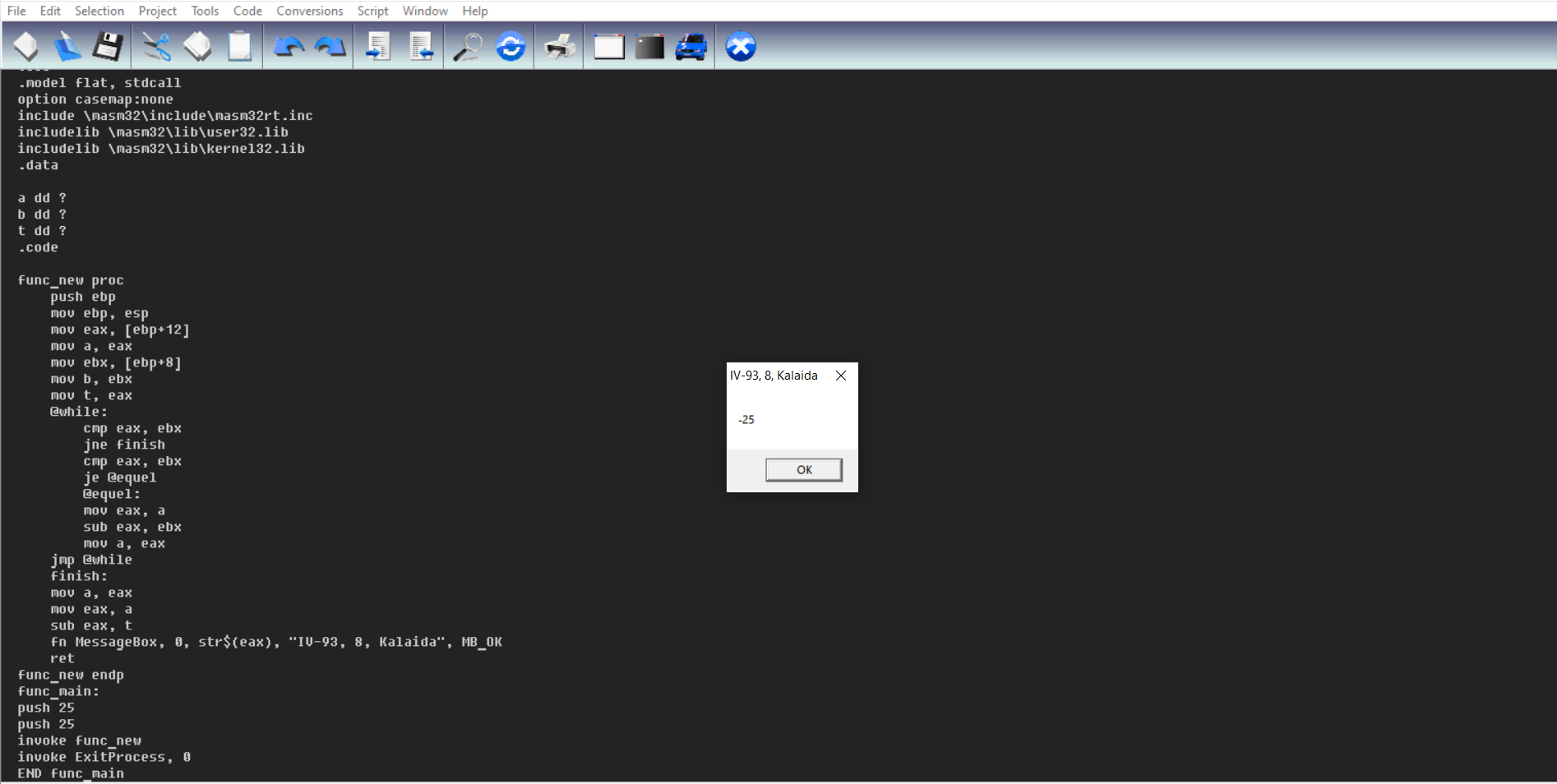
****

Код masm32, який був згенерований генератором кода



Результат

****

****

**Лістинг програми**

Text.py – файл, в якому знаходиться вхідний код на мові Python

Lexer.py – клас лексера;

Parser.py – клас парсера;

Tree.py – функція, яка генерує дерево;

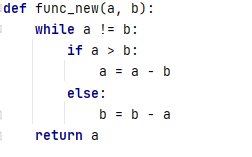
Start.py – запуск всіх класів;

main.py – файл, який запускає програму;

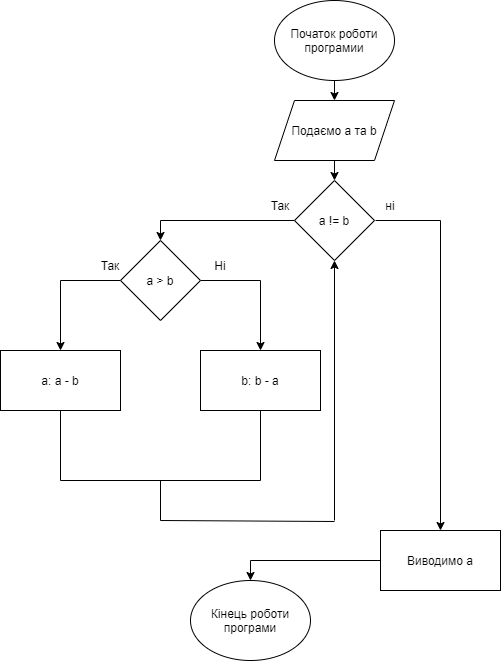
**Виконання завдання**

За варіантом №8 мені потрібно було реалізувати алгоритм №8 та вхідний код повинен бути на мові програмування Python. Алгоритм №8 – реалізувати алгоритм Евкліда для знаходження НСК та НСД. На вхід подається невід`ємне число.

Для знаходження НСД мною був розроблений алгоритм для мові програмування Python.

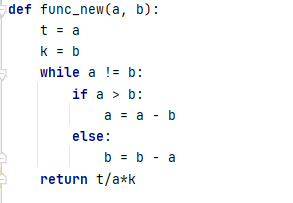
**

**Алгоритм знаходження найбільшого спільного дільник**а

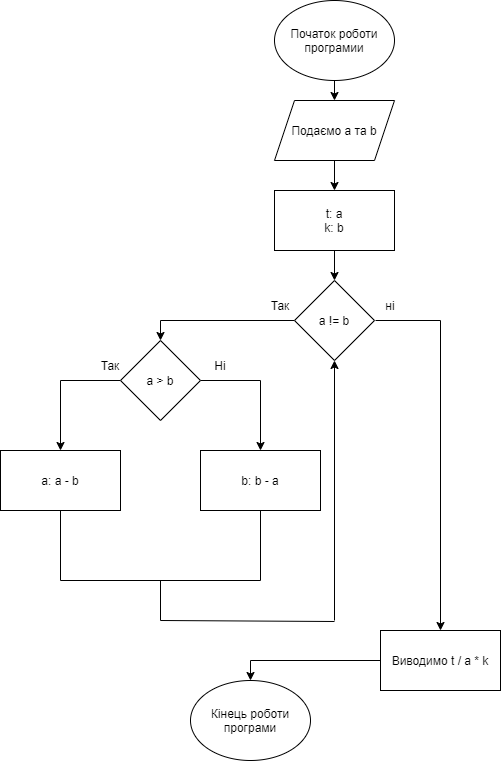


1. Input a, b
2. If a == b, go to point 5
3. If: a > b: a = a – b, else: b = b – a
4. Go to point 2
5. Get result a

Для знаходження НСК мною був розроблений алгоритм для мові програмування Python.



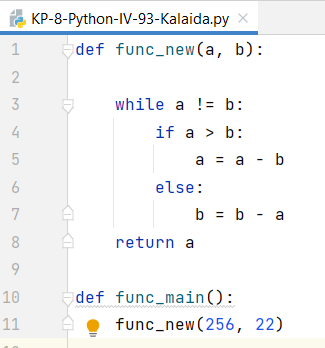
**Алгоритм знаходження найбільшого спільного Кратного**



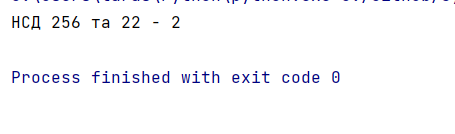
1. Input a, b
2. t: a, k: b
3. If a == b, go to point 6
4. If: a > b: a = a – b, else: b = b – a
5. Go to point 3
6. Get result t / a \* k

**Перевірка роботи програми (НСД)**

1. Вводимо 2 числа: 256 і 22 (НСД)

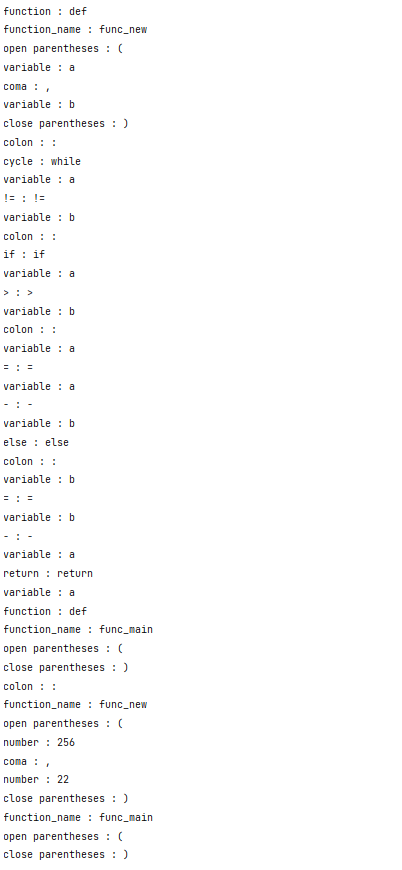


Перевірка за допомогою python



Відповідь: 2

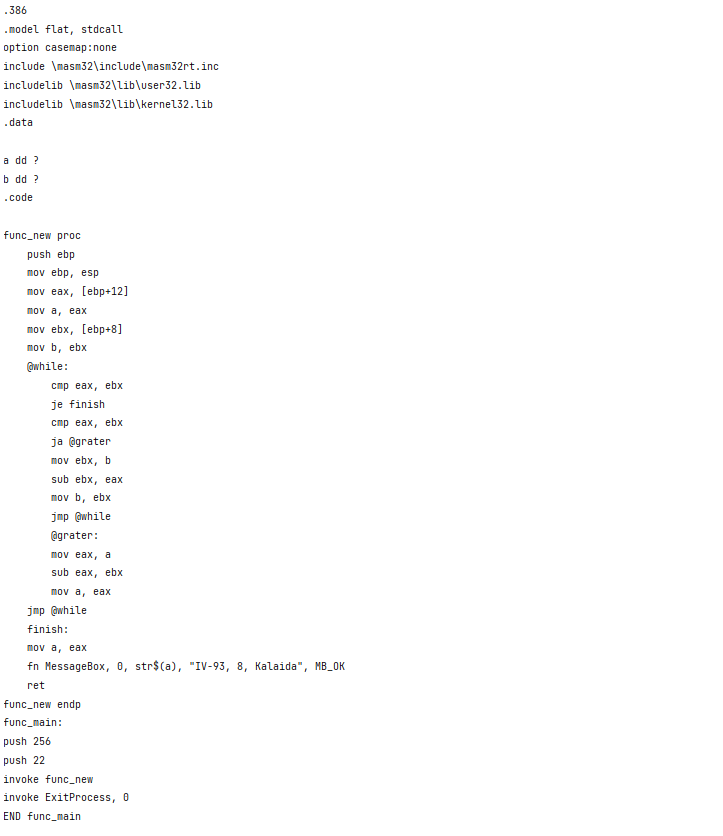
Токени



Дерево



Вихідний код на masm32

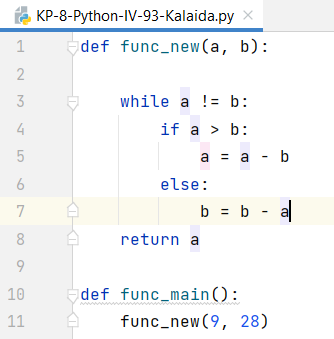


Перевірка в masm32

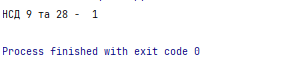


Все вірно

1. Вводимо 2 числа: 9 і 28 (НСД)

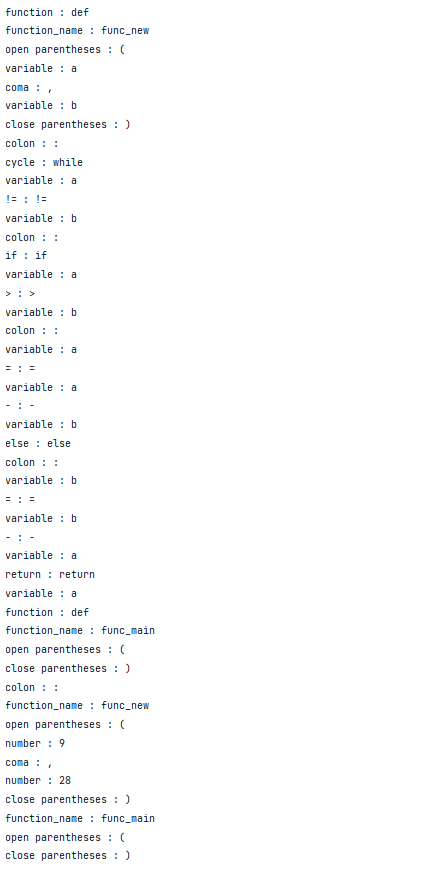


Перевірка за допомогою python



Відповідь: 1

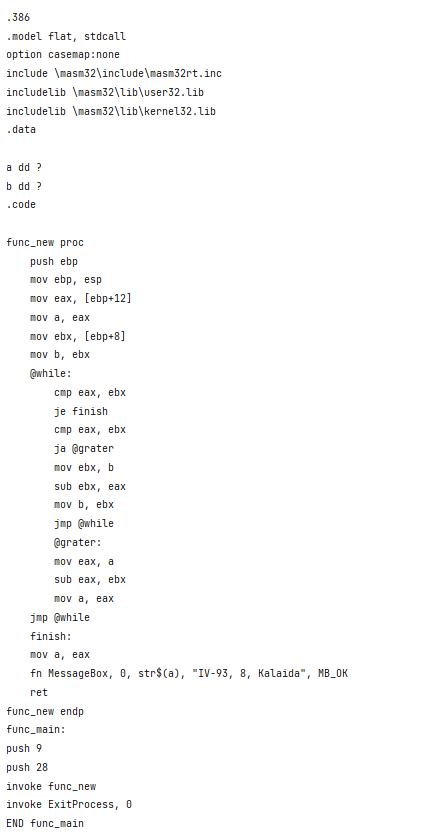
Токени



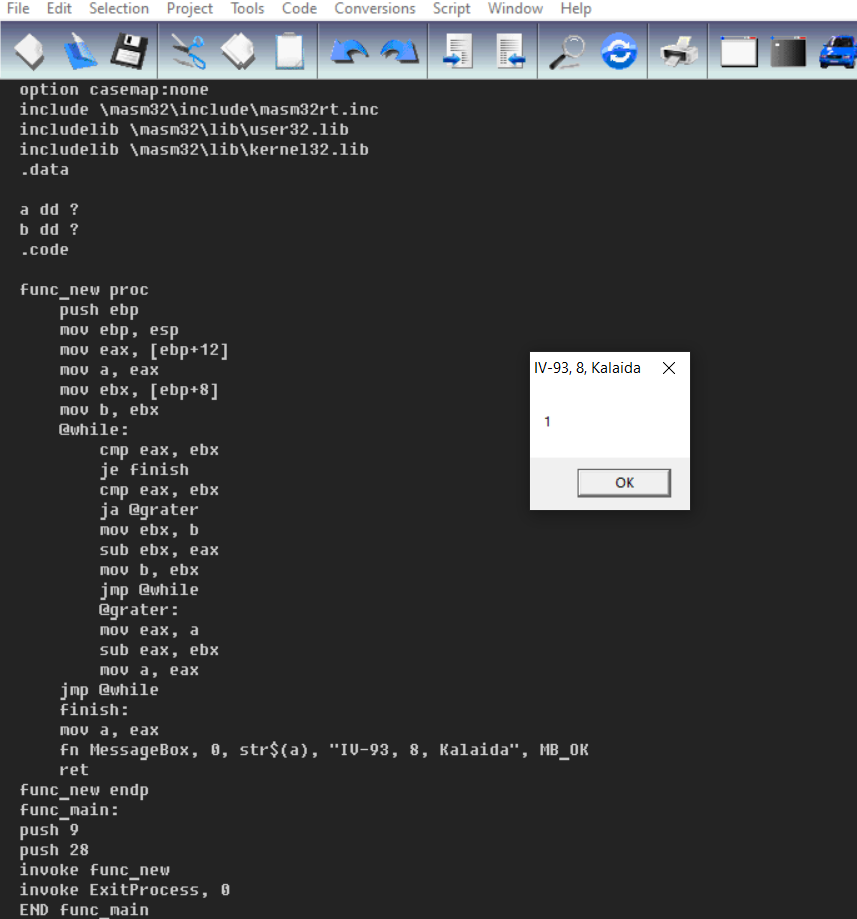
Дерево



Вихідний код на masm32

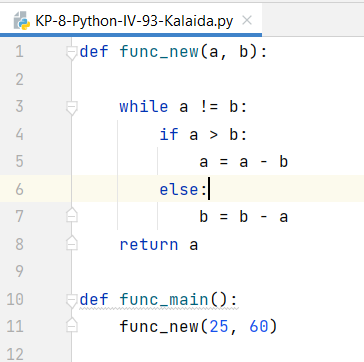


Перевірка в masm32

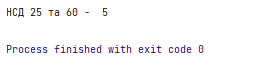


Все вірно

1. Вводимо 2 числа: 25 і 60 (НСД)

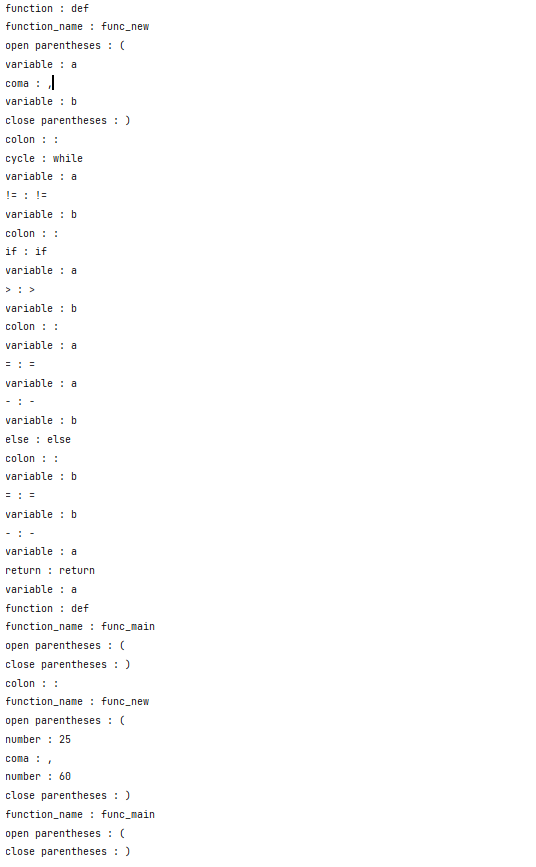


Перевірка за допомогою python



Відповідь: 5

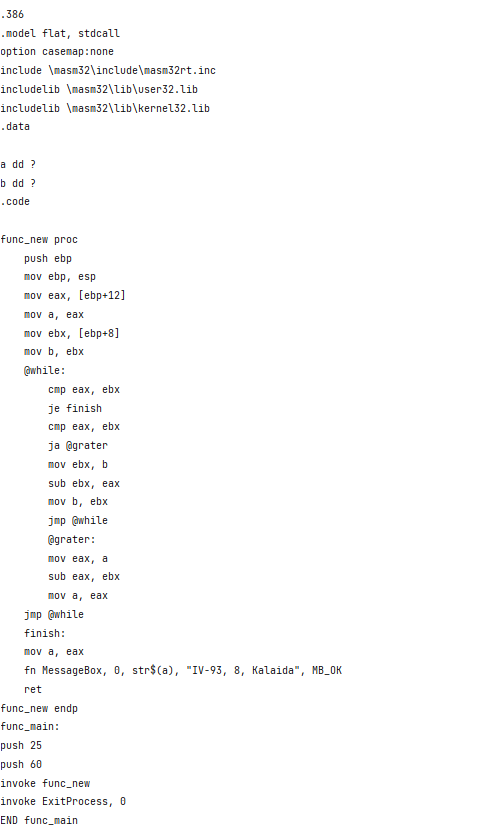
Токени



Дерево



Код на masm32

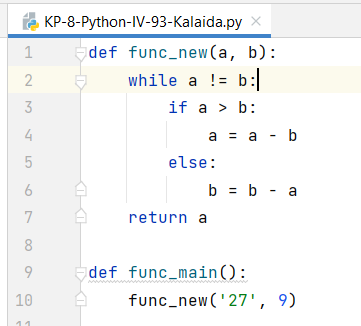


Перевірка в masm32

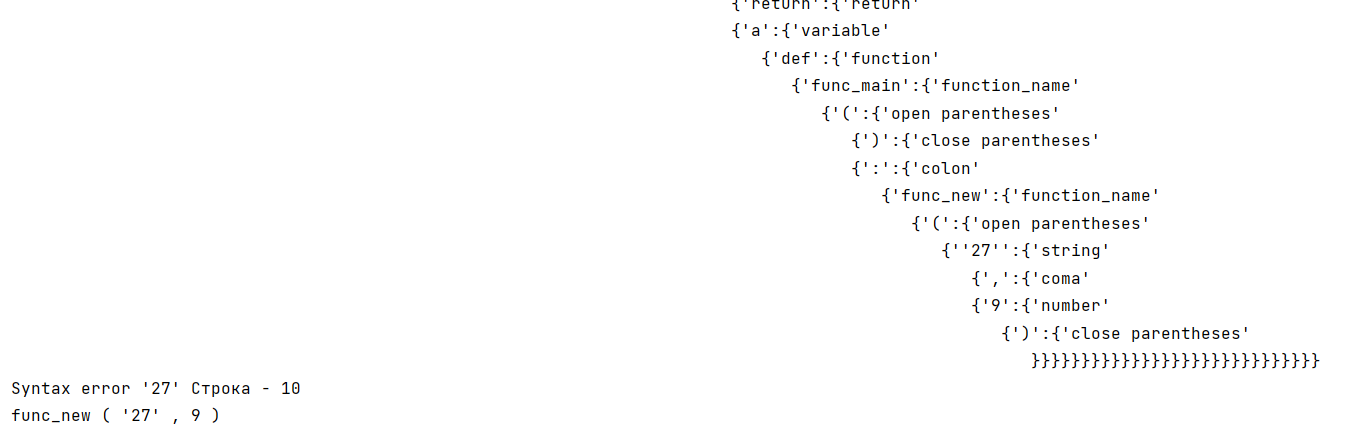


Все вірно

1. Вводимо строку та число: “27” та 9 (НСД)

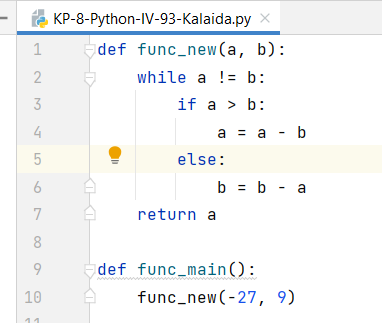
****

Результат

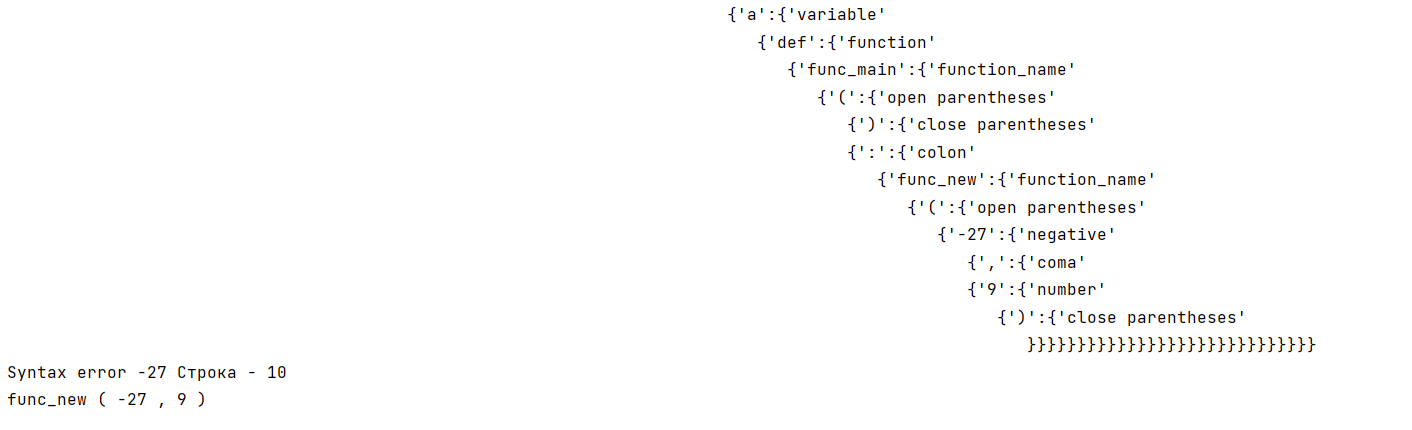


Помилка

1. Вводимо від`ємне число та додатне число: -27 та 9 (НСД)



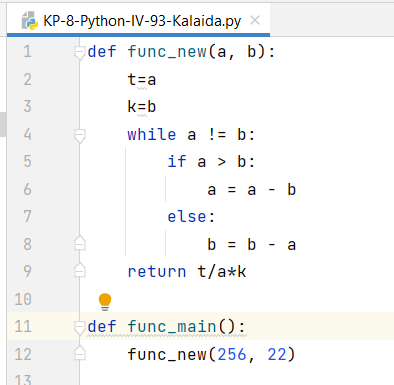
Результат



Помилка

**Перевірка роботи (НСК)**

1. Вводимо 2 числа: 256 і 22 (НСК)

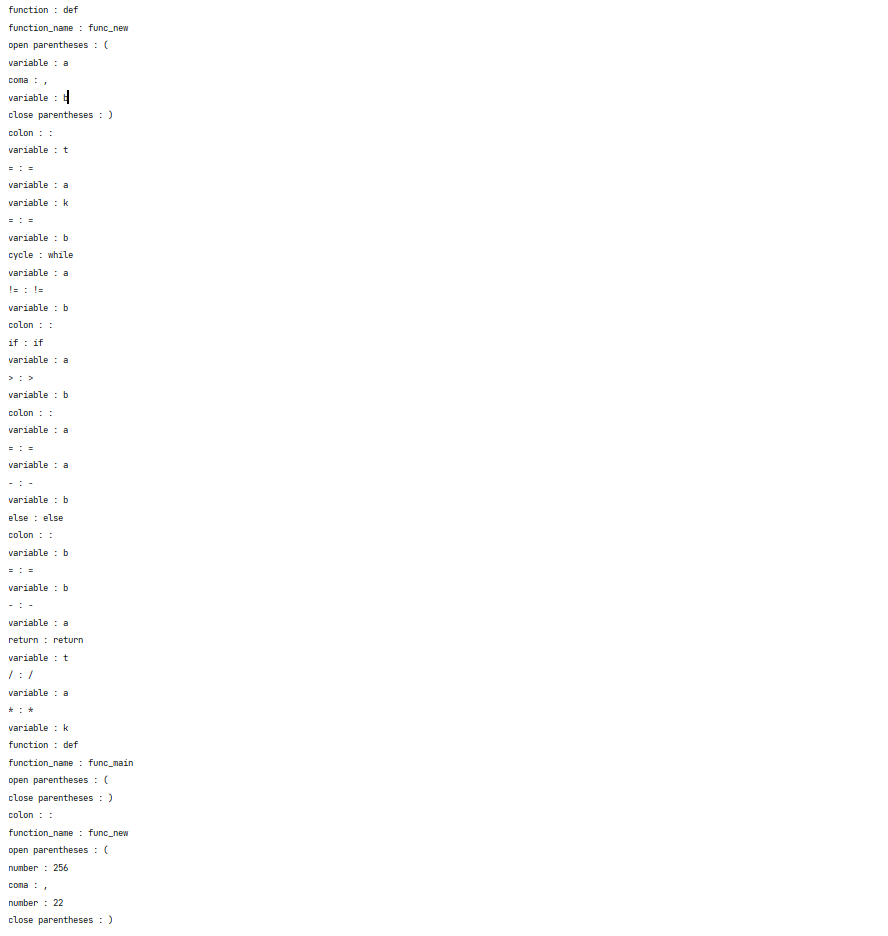


Перевірка за допомогою python



Відповідь: 2816

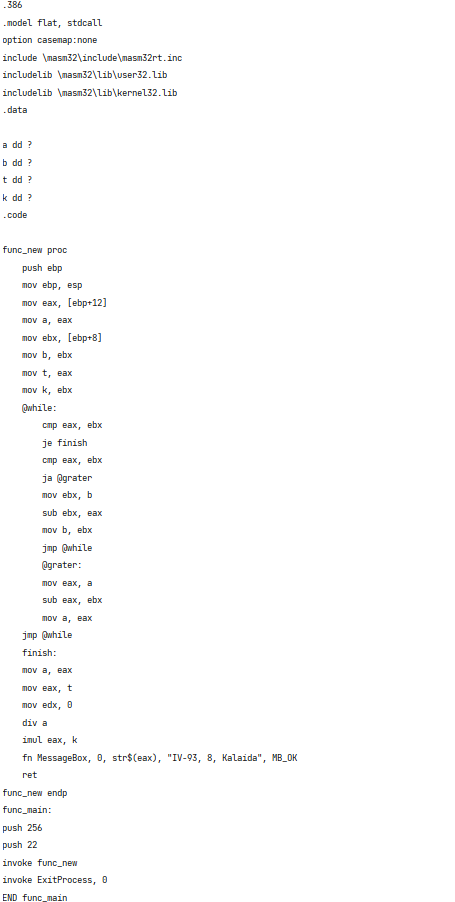
Токени



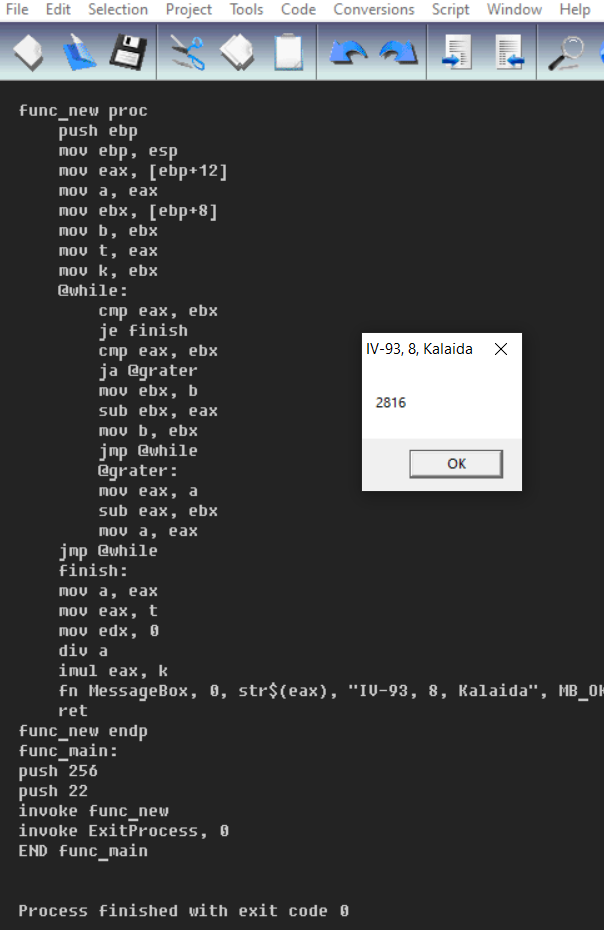
Дерево



Код для masm32

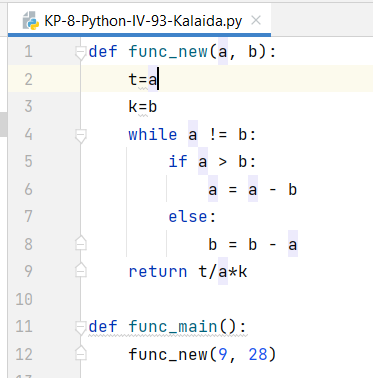


Перевірка в masm32

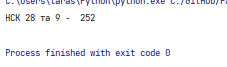


Все вірно

1. Вводимо 2 числа: 9 і 28 (НСК)

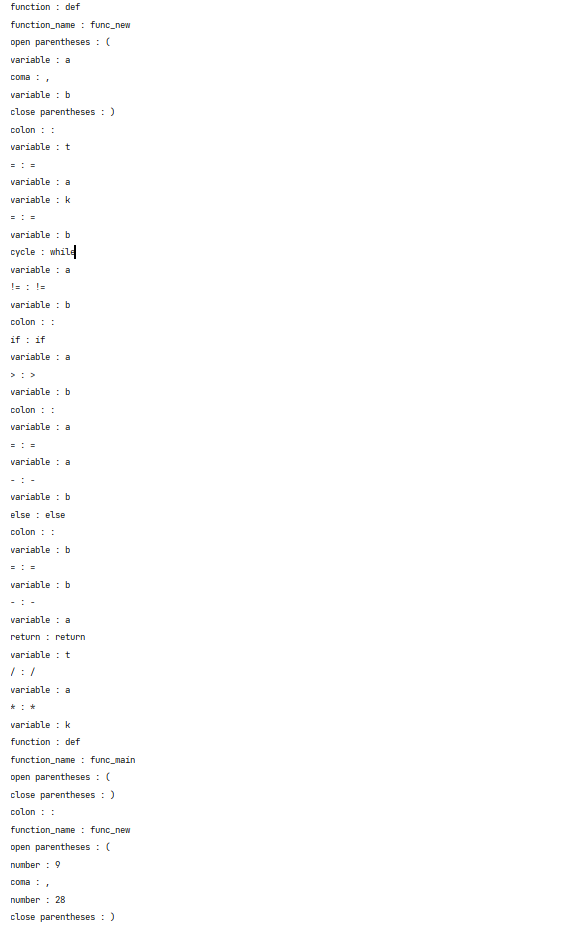


Перевірка за допомогою python



Отриманий результат: 252

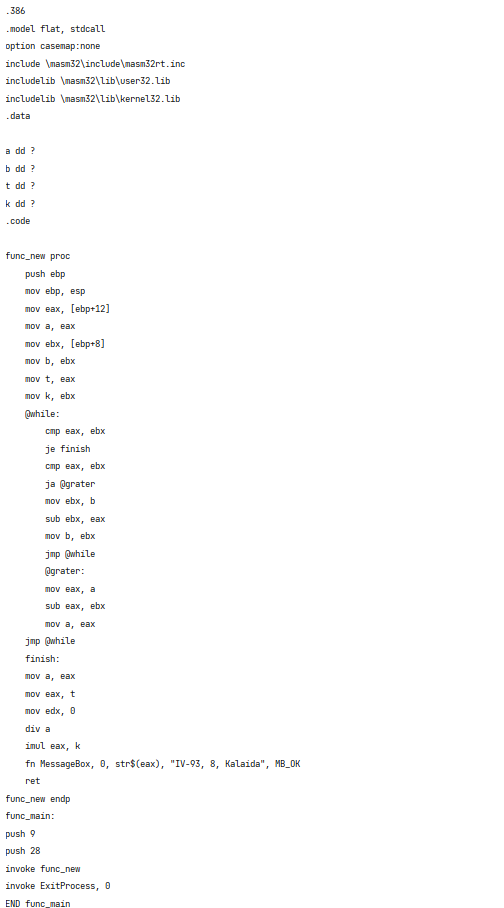
Токени



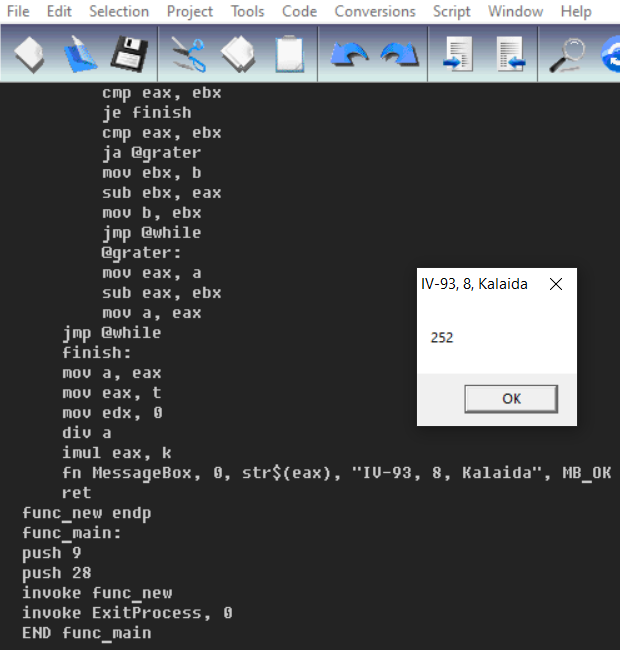
Дерево



Вихідний код на masm32

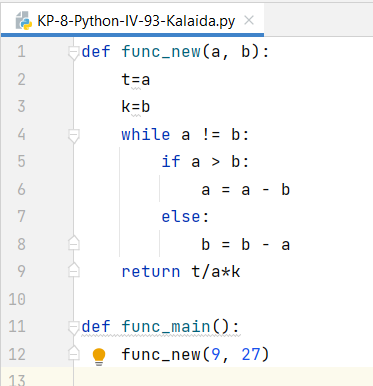


Перевірка в masm32



Все вірно

1. Вводимо 2 числа: 9 і 27 (НСК)

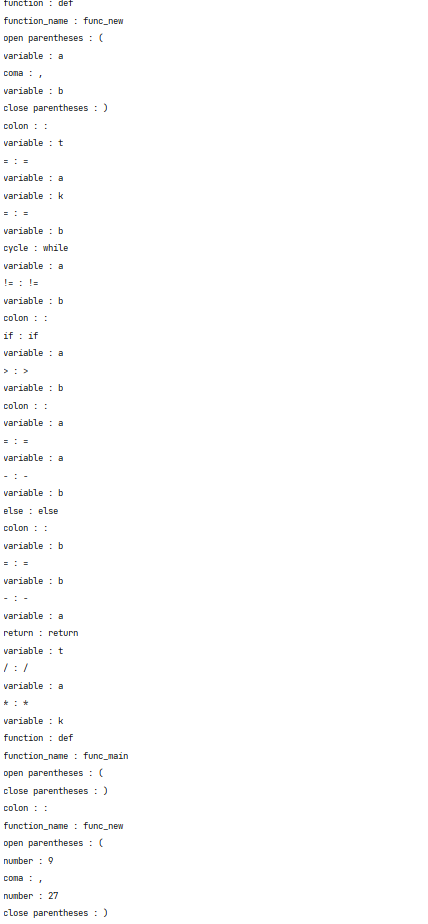


Перевірка за допомогою python



Отриманий результат: 27

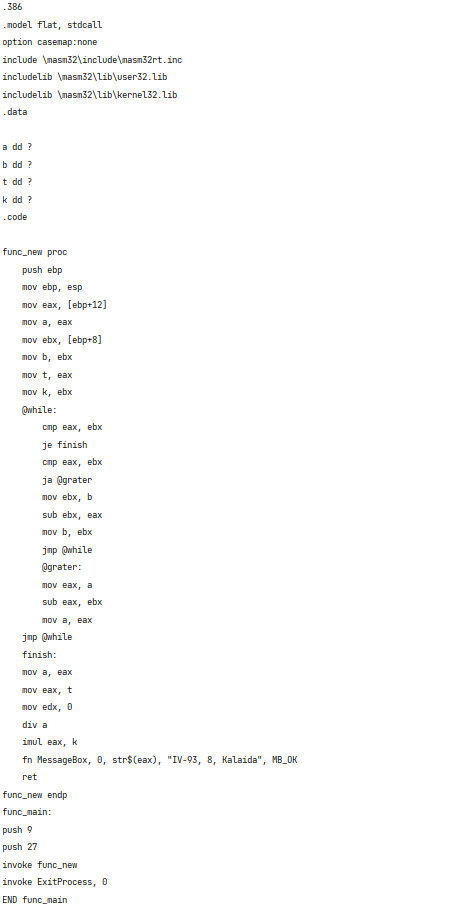
Токени



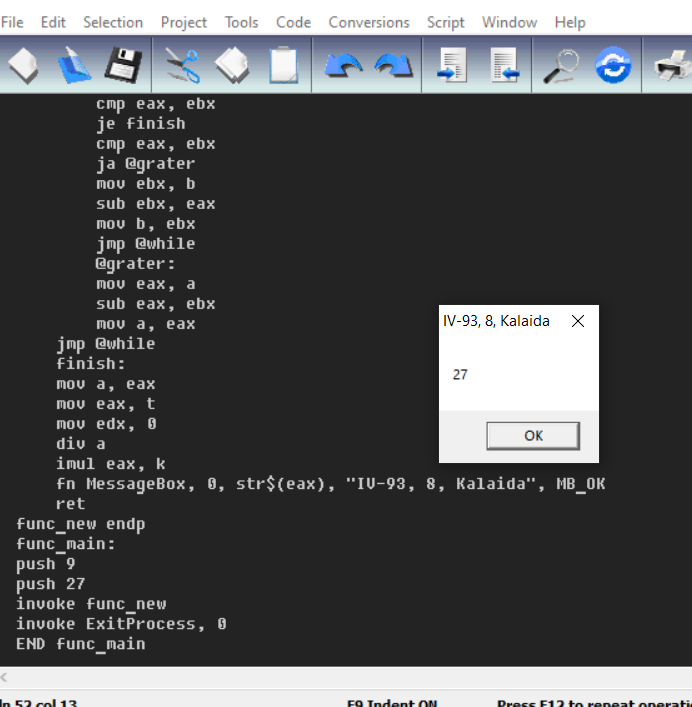
Дерево



Код на masm32

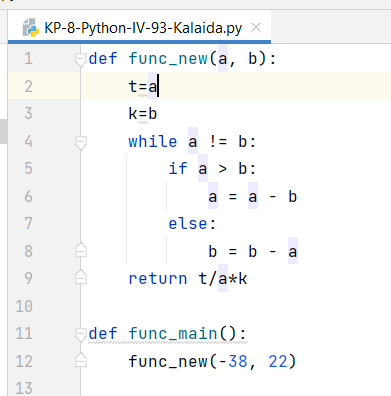


Перевірка в masm32

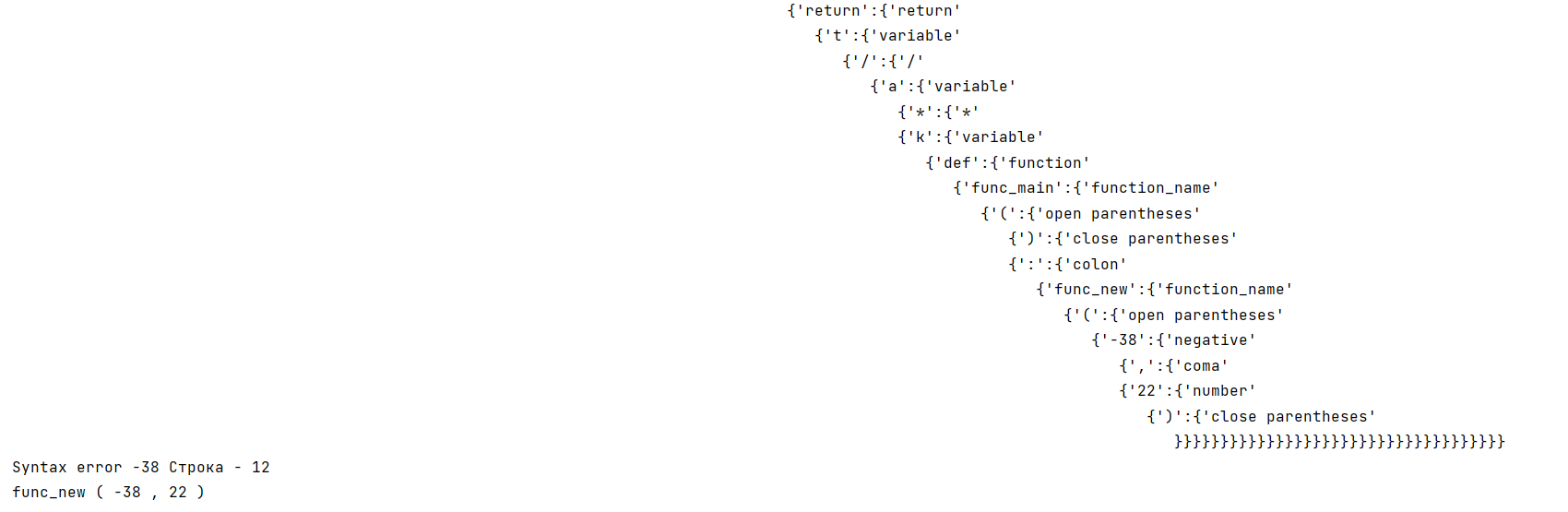


Все вірно

1. Вводимо від`ємне число та додатне число: -38 та 22 (НСК)

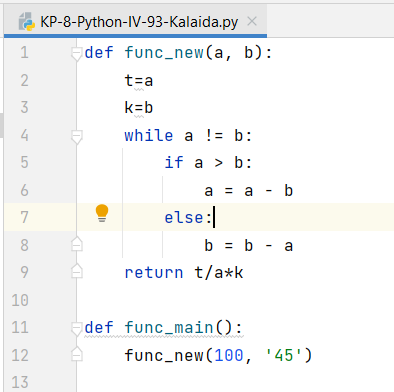


Результат

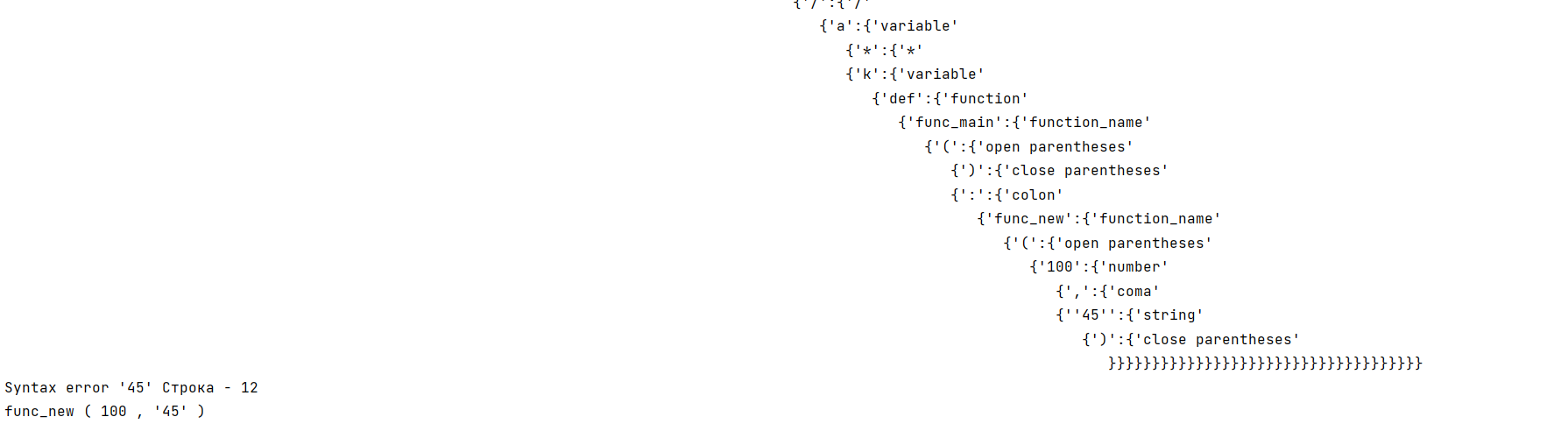


Помилка

1. Вводимо строку та число: ‘45’ та 100



Результат



Помилка

**Висновки**

Я результаті курсової робити я отримав неймовірно корисні та важливі навички з побудови компіляторів. Компілятор був розроблений на достатньо сучасній мові програмування Python. Я думаю, що всі, зароблені під час цієї роботи, навички я зможу використати у своїй кар`єрі.

Я вважаю, що робота була виконана успішно так без складних проблем. Було створено лексер, парсер та генератор коду. В результаті отримано masm32 файли, які були перевірені. За варіантом я мав реалізувати алгоритм Евкліда (НСК та НСД), що я і зробив. Створений мною скрипт простий у розумінні, але при вводі великих чисел, можуть використовуватися значні ресурси комп`ютера.

Також було виконано обробку помилок для захисту від некоректного вводу, тобто для від`ємних чисел та строк

**Використані джерела**

1. Методичка для лабораторних робіт курсу «Системне програмування – 2», Павлов В.Г. 2020.
2. https://uk.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC\_%D0%95%D0%B2%D0%BA%D0%BB%D1%96%D0%B4%D0%B0
3. <http://py-algorithm.blogspot.com/2011/04/blog-post_03.html>
4. https://foxford.ru/wiki/informatika/algoritm-evklida-python
5. http://ua.onlinemschool.com/math/assistance/number\_theory/nod\_nok/