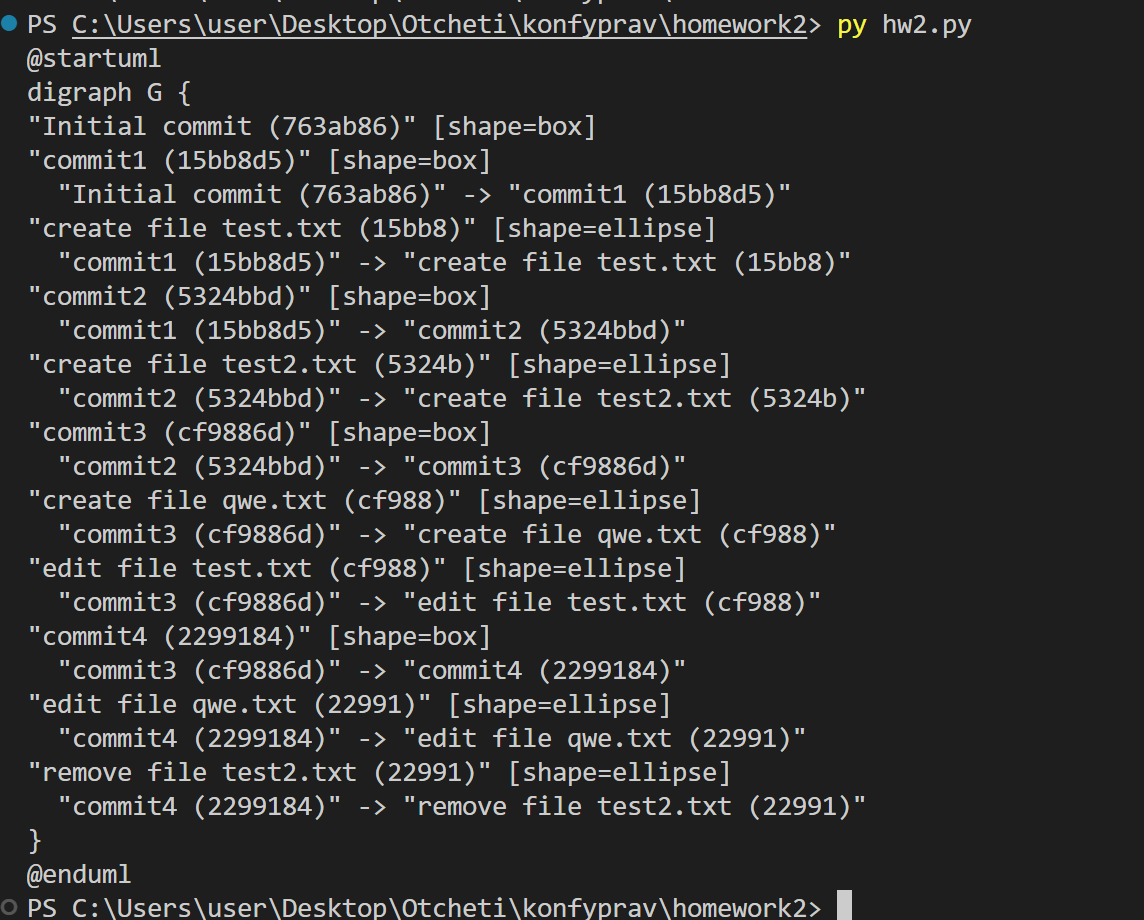
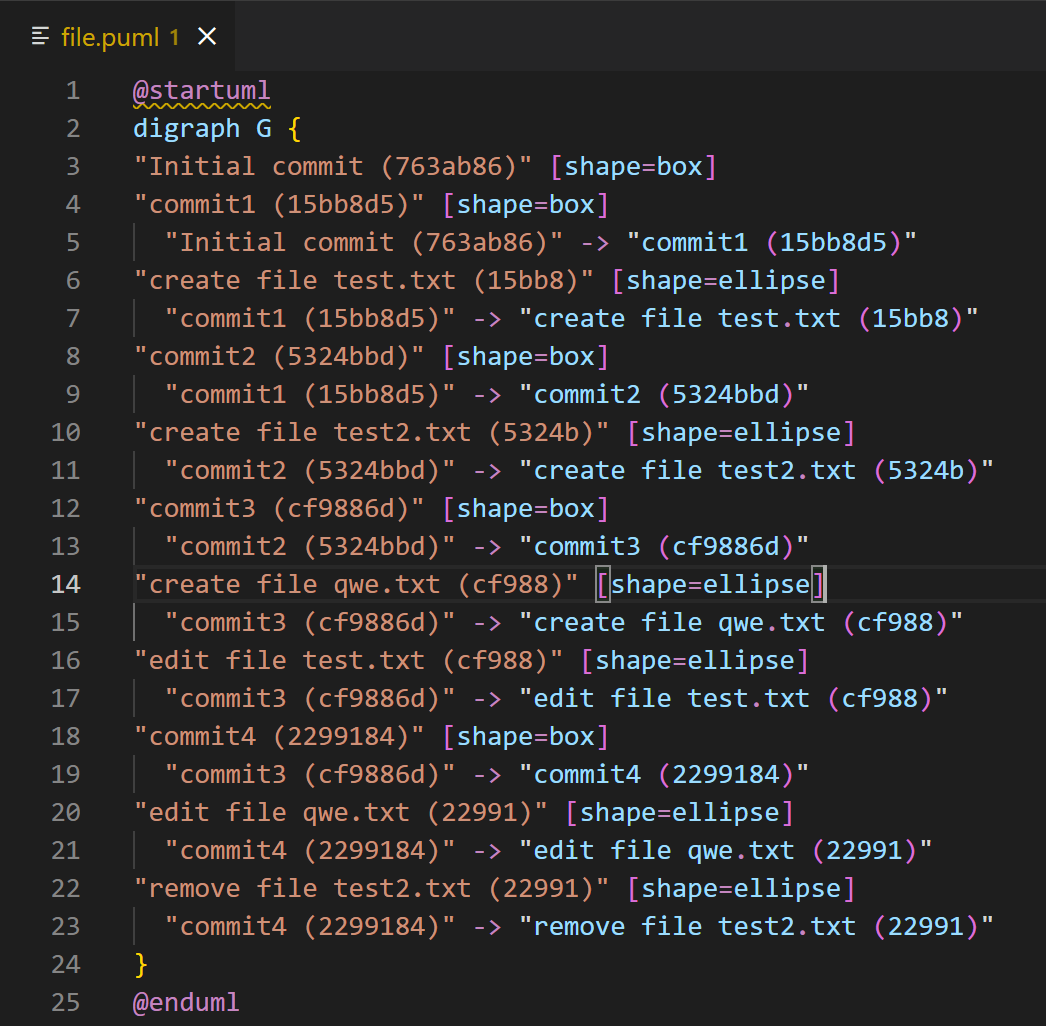
Код программы

|  |
| --- |
| import subprocess  import xml.etree.ElementTree as ET  import os  def read\_config(config\_path):      tree = ET.parse(config\_path)      root = tree.getroot()      config = {          'visualizer\_path': root.find('visualizer\_path').text,          'repo\_path': root.find('repo\_path').text,          'output\_path': root.find('output\_path').text      }      return config  def get\_commit\_tree(repo\_path):      result = subprocess.run(          ['git', '-C', repo\_path, 'log', '--pretty=format:%H %s', '--reverse'],          stdout=subprocess.PIPE,          stderr=subprocess.PIPE,          text=True,          encoding='utf-8'      )        if result.returncode != 0:          print("Ошибка при получении коммитов:", result.stderr)          return {}      commits = result.stdout.splitlines()      commit\_info = {} #инфа для каждого коммита        for commit in commits:          hash\_value, message = commit.split(' ', 1)          commit\_info[hash\_value] = {              'message': message,              'children': []          }      # информация о дереве файлов для каждого коммита      for hash\_value in commit\_info.keys():          result = subprocess.run(              ['git', '-C', repo\_path, 'diff-tree', '--no-commit-id', '--name-status', '-r', hash\_value],              stdout=subprocess.PIPE,              stderr=subprocess.PIPE,              text=True,              encoding='utf-8'          )            if result.returncode != 0:              print(f"Ошибка при получении файлов для коммита {hash\_value}:", result.stderr)              continue            files = result.stdout.splitlines()            for file in files:              status, filename = file.split('\t')              commit\_info[hash\_value]['children'].append((status, filename))        return commit\_info  def generate\_plantuml\_code(commit\_info): #создание дерева коммитов      lines = ["@startuml", "digraph G {"]        commit\_nodes = {}        previous\_commit = None        for commit\_hash, info in commit\_info.items():          short\_commit\_hash = commit\_hash[:7]  #показываем хэш коммита          commit\_node = f'"{info["message"]} ({short\_commit\_hash})" [shape=box]'          lines.append(commit\_node)          commit\_nodes[commit\_hash] = info["message"]          if previous\_commit:              lines.append(f'  "{previous\_commit}" -> "{info["message"]} ({short\_commit\_hash})"')          for status, filename in info['children']:              action\_hash = commit\_hash[:5]  #показываем хэш действия                if status == 'A':                  action = f'create file {filename} ({action\_hash})'              elif status == 'M':                  action = f'edit file {filename} ({action\_hash})'              elif status == 'D':                  action = f'remove file {filename} ({action\_hash})'              else:                  action = f'unknown action {filename} ({action\_hash})'              file\_node = f'"{action}" [shape=ellipse]'              lines.append(file\_node)              lines.append(f'  "{info["message"]} ({short\_commit\_hash})" -> "{action}"')          previous\_commit = f"{info['message']} ({short\_commit\_hash})"      lines.append("}")      lines.append("@enduml")        return "\n".join(lines)    def write\_output(output\_path, content):      with open(output\_path, 'w') as f:          f.write(content)  def main(config\_path):      config = read\_config(config\_path)      commit\_info = get\_commit\_tree(config['repo\_path'])      plantuml\_code = generate\_plantuml\_code(commit\_info)      print(plantuml\_code)      write\_output(config['output\_path'], plantuml\_code)  if \_\_name\_\_ == "\_\_main\_\_":      import sys      if len(sys.argv) != 2:          print("Напишите: py hw2.py <config\_path>")          sys.argv.append("config.xml")      main(sys.argv[1]) |

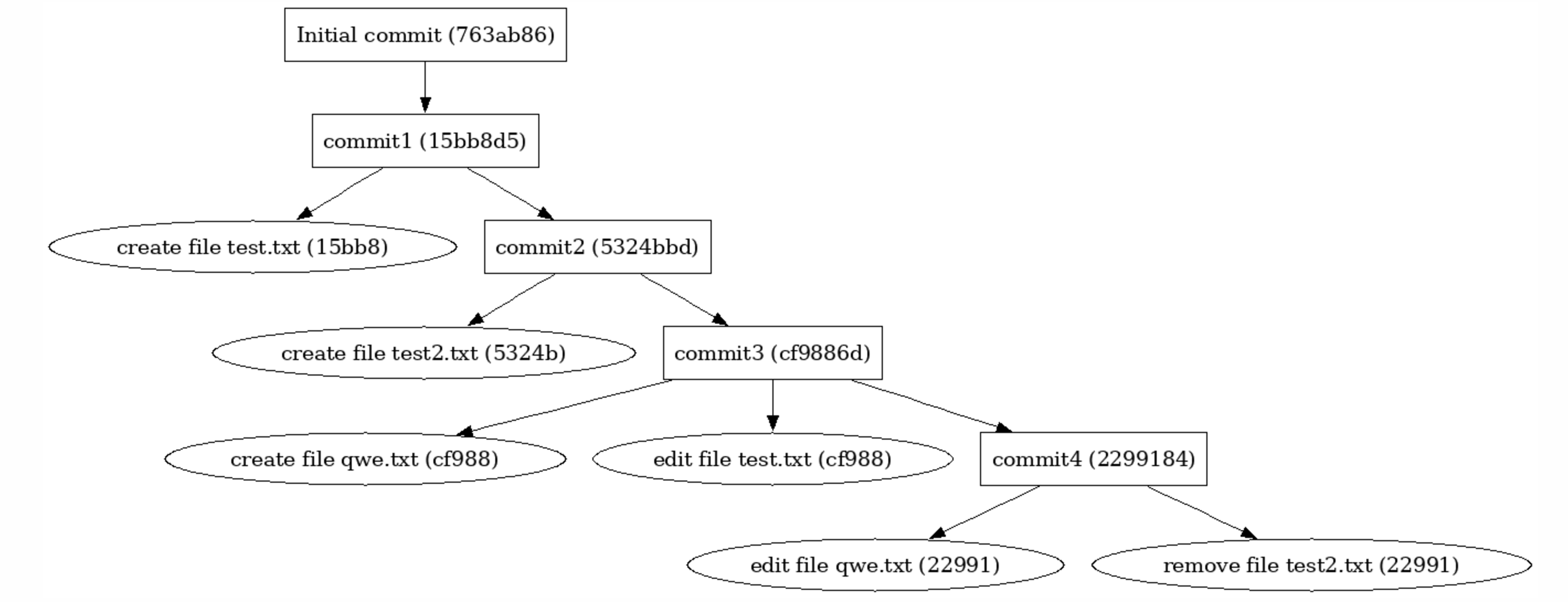
Запуск и вывод программы



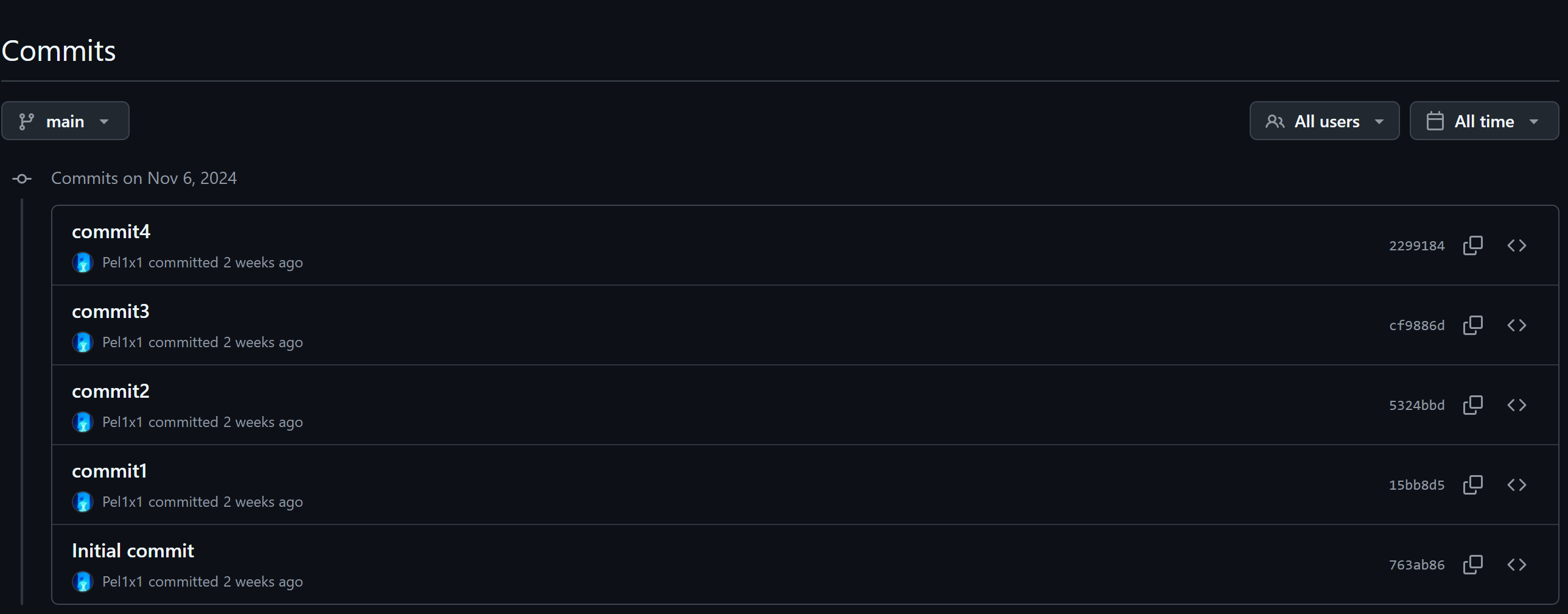
Файл .puml для построение дерева



Построенное дерево



История коммитов на GitHub



Отработка unit тестов

