**Домашняя работа 5**

**Задание 1:**

Описание Разработать CLI-утилиту для выдачи информации по локальному git-репозиторию. Назначение утилиты – визуализатор графа изменений дерева (tree: файлы и папки) проекта с выводом в формате dot (graphviz).

Данные извлекаются из папки .git без использования сторонних программ и библиотек. То есть нельзя из программы вызывать команды git для получения данных (например "git log" и "git cat-file"). Но можно и нужно пользоваться сведениями отсюда: <https://git-scm.com/book/ru/v2/Git-%D0%B8%D0%B7%D0%BD%D1%83%D1%82%D1%80%D0%B8-%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D1%8B-Git>

Обязательно приведите примеры работы вашей утилиты.

|  |
| --- |
| import sys  import random  from git import Repo  import graphviz  def generate\_hex\_color():  red = random.randint(0, 255)  green = random.randint(0, 255)  blue = random.randint(0, 255)  hex\_color = '#%02x%02x%02x' % (red, green, blue)  return hex\_color  def main(args):  if len(args) == 1:  print("Please, enter abs path to repo")  sys.exit(1)  dot = graphviz.Digraph('Graph')  path = args[1]  # Получаем список объектов Git  repo = Repo(path, search\_parent\_directories=True)  # Добавляем узел для ветки  branch\_node = f"Branch\_{repo.active\_branch.name}"  dot.node(branch\_node, label=f"Branch: {repo.active\_branch.name}", shape='box')  # Добавляем узлы и рёбра для каждого коммита и файла  for commit in repo.iter\_commits():  commit\_header = commit.message.split('\n')[0]  color = generate\_hex\_color()  node\_id = str(commit)  # Check if the commit is the HEAD commit  if node\_id == str(repo.head.commit):  dot.node(node\_id, label=f'{commit.author}, date: {commit.authored\_date}, header: {commit\_header}, hash: {commit.hexsha}', color='orange')  else:  dot.node(node\_id, label=f'{commit.author}, date: {commit.authored\_date}, header: {commit\_header}, hash: {commit.hexsha}')    # Add edge from branch node to commit node  dot.edge(branch\_node, node\_id, color=color)  for el in commit.tree.traverse():  if el.type == 'blob':  blob\_sha = el.hexsha  contents = repo.git.cat\_file('-p', blob\_sha)  dot.node(str(el.hexsha) + str(commit), label=f'{el.path}\nhash: {blob\_sha}\ncontents: {contents}')  dot.edge(node\_id, str(el.hexsha) + str(commit), color=color)  # Set node attributes  dot.node\_attr["shape"] = "ellipse"  # Set graph attributes  dot.graph\_attr["rankdir"] = "LR"  dot.attr(nodesep="2")  dot.attr(ranksep="20")  dot.attr(fontsize="20")  dot.render("repo", view=True, format='svg')  if \_\_name\_\_ == "\_\_main\_\_":  main(sys.argv) |

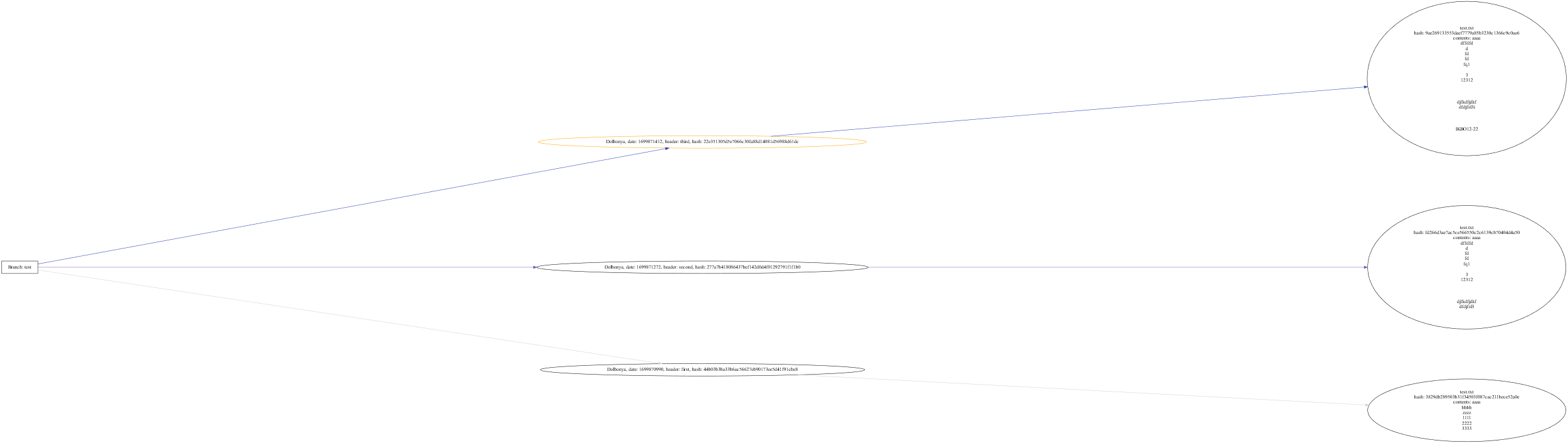


Рисунок 1 - Пример работы программы