**Домашняя работа 2**

**Задание 1:**

Написать на выбранном вами языке программирования программу, которая принимает в качестве аргумента командной строки имя пакета, а возвращает граф его зависимостей в виде текста на языке Graphviz. На выбор: для npm или для pip. Пользоваться самими этими менеджерами пакетов запрещено. Главное, чтобы программа работала даже с неустановленными пакетами и без использования pip/npm.

|  |
| --- |
| *# Импорт необходимых модулей* import os *# Для работы с операционной системой* import sys *# Для работы с параметрами командной строки* import requests *# Для отправки HTTP-запросов* import graphviz as gv *# Для создания графов* import pydot *# Для работы с DOT-языком и графами  # Функция для чтения аргументов командной строки* def read\_args():  args = []  for arg in sys.argv[1:]: *# Перебор аргументов, начиная со второго (первый - имя скрипта)* args.append(arg)  return args  *# Функция для получения зависимостей пакета* def get\_dependencies(package\_name) -> list:  url = 'https://pypi.org/pypi/{}/json'  json = requests.get(url.format(package\_name)).json()  try:  requirements = json['info']['requires\_dist']  except:  return []  data = []  if requirements is not None:  for req in requirements:  data.append(req.split(' ')[0])  return data  deptree = []  def parse\_deps(deps: list, first: bool = True):  print(deps.\_\_len\_\_())   if (deps.\_\_len\_\_() != 0):  for dep in deps:  deptree.append((dep, get\_dependencies(dep)))  def recursiveNodes(graph, name, depth=0):  deps = get\_dependencies(name)   if (deps.\_\_len\_\_() == 0 or depth > 1):  return   for dep in deps:  graph.node(dep, dep)  graph.edge(name, dep)  recursiveNodes(graph, dep, depth + 1)  *# Главная функция* def main(args):  if len(args) == 0:  print("Не указано название пакета")  return   dot = gv.Digraph(comment='Зависимости {}'.format(args[0])) *# Создание графа с комментарием* dot.node(args[0], args[0]) *# Добавление узла с именем первого пакета* recursiveNodes(dot, args[0])   dot.render('output/table.gv', view=True)  (graph,) = pydot.graph\_from\_dot\_file('output/table.gv')  graph.write\_png('output/table.png')  if \_\_name\_\_ == "\_\_main\_\_":  main(read\_args()) |