Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Факультет прикладной математики и механики

Кафедра «Вычислительная математика, механика и биомеханика»

Направление: 09.03.02 «Информационные системы и технологии»

**ЛАБОРАТОРНАЯ РАБОТА № 2**

**Проектирование и реализация оболочки экспертной**

**системы, использующей продукционный способ представления знаний**

Выполнил:

студент гр. ИСТ-19-2Б, А.С. Бабикова

Проверил:

ассистент каф. ВММБ, Г.В. Нетбай

Пермь 2022

Описание средств решения

### Для реализации поставленной задачи был выбран язык высокого уровня Python. Формат хранения базы знаний – JSON.

Описание решаемой задачи

### Программа состоит из 9 функций:

### Главная (run);

### Две функции с помощью, которой задаются вопросы пользователю (yes\_or\_no, question\_weather);

### Функция, отвечающая за вывод информации в консоль (pprint);

1. Функция сравнения (sravn);
2. Функция логического И (op\_and);
3. Функция get;
4. Функция set, которая сохраняет ответы пользователя в память (memory);
5. Функция found, которая устанавливает переменную solution = 1, что означает, что решение найдено.

Рассмотрим подробнее работу каждой функции в отдельности.

1. На рис. 1 представлена функция run.

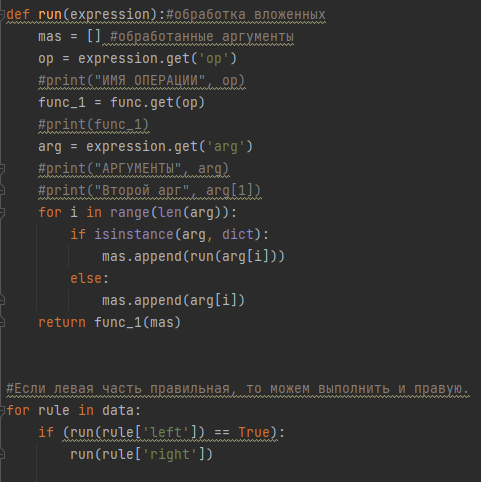


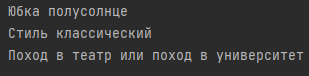
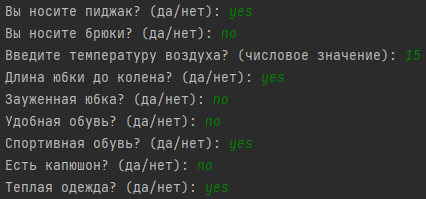
Рисунок 1 – Функция run.

### Данная функция работает с данными файла json, в котором прописана вся логика программы, то есть определяет, что за функцию требуется вызвать и с какими аргументами.

Так же есть цикл, который проверяет условие, что если левая часть правильная, то можем выполнить и правую.

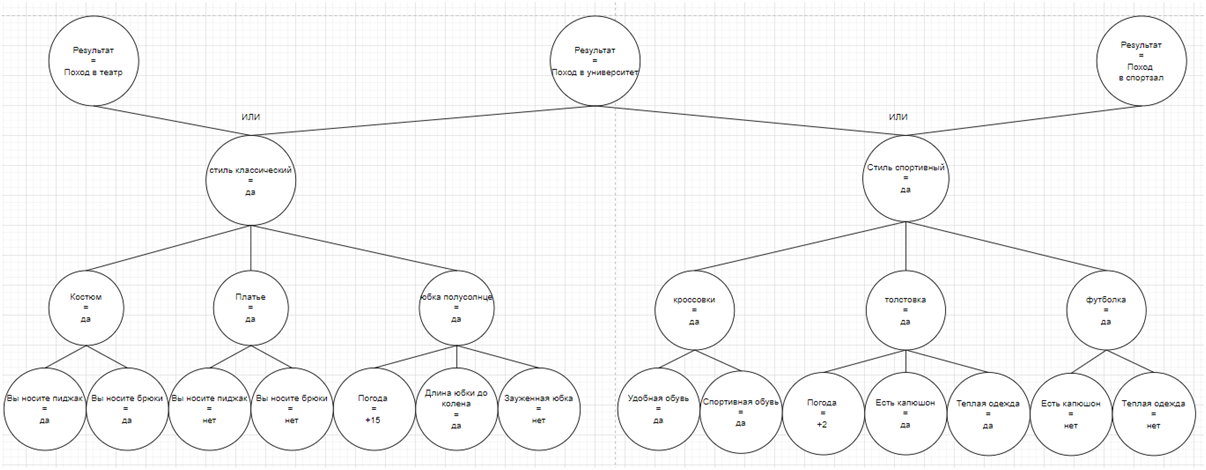
2. Остальные функции являются вспомогательными, вызываются, когда функция run перебирает данные из файла json.

3. На рис.3 можно увидеть работу программы.



### Рисунок 3 – Результат работы программы.

### Сравнивая с графом, программа работает правильно.



Вывод

### Была реализована оболочка ЭС, которая позволяет работать с продукционной базой знаний.

Приложение

import json  
  
with open('base.json', 'r', encoding='utf-8') as fh: # открываем файл на чтение  
 data = json.load(fh) # загружаем из файла данные в словарь data  
 print("База знаний", data)  
  
  
def sravn(arg):  
 x = arg[0]  
 if x in memory:  
 if(arg[1]==memory[x]):  
 return False  
 return True  
  
  
def op\_and(arg):  
 k = 0  
 for i in range(0, len(arg)):  
 if run(arg[i]) == False :  
 k += 1  
 if k == len(arg):  
 return True  
 if k > 0:  
 return False  
  
  
def pprint(arg):  
 arg = ''.join(arg)  
 print(arg)  
  
  
  
memory = {"solution": 0, "jacket": 0, "trousers": 0, "weather": 0, "length": 0, "tight\_skirt": 0, "comfortable": 0, "sport": 0, "hood": 0, "warm": 0}  
  
  
def get(arg):  
 return memory[arg[0]]  
  
  
def yes\_or\_no(arg):  
 arg = ''.join(arg)  
 answer = input(arg + ' (да/нет): ')  
 return answer  
  
  
def question\_weather(arg):  
 arg = ''.join(arg)  
 answer = int(input(arg + ' (числовое значение): '))  
 return answer  
  
  
def op\_set(arg):  
 x = arg[0]  
 for i in range(1, len(arg)):  
 res = run(arg[i])  
 if x in memory:  
 if (res != None):  
 memory[x] = res  
  
  
def found(arg):  
 x = arg[0]  
 if x in memory:  
 memory[x] = 1  
  
   
   
func = {'==': sravn,  
 'print': pprint,  
 'get': get,  
 'yes\_or\_no': yes\_or\_no,  
 'question\_weather': question\_weather,  
 'and': op\_and,  
 'set': op\_set,  
 'found': found  
 }  
  
def run(expression):#обработка вложенных  
 mas = [] #обработанные аргументы  
 op = expression.get('op')  
 #print("ИМЯ ОПЕРАЦИИ", op)  
 func\_1 = func.get(op)  
 #print(func\_1)  
 arg = expression.get('arg')  
 #print("АРГУМЕНТЫ", arg)  
 #print("Второй арг", arg[1])  
 for i in range(len(arg)):  
 if isinstance(arg, dict):  
 mas.append(run(arg[i]))  
 else:  
 mas.append(arg[i])  
 return func\_1(mas)  
  
  
#Если левая часть правильная, то можем выполнить и правую.  
for rule in data:  
 if (run(rule['left']) == True):  
 run(rule['right'])