**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования** **«Новгородский государственный университет имени Ярослава Мудрого»**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра ИТиС

«Поисковая система на основе фреймовой модели представления знаний»

Лабораторная работа №3

По дисциплине: Системы искусственного интеллекта

Отчет

Выполнил: студент гр. 8091

\_\_\_\_\_\_\_ \_\_\_\_Ефимов А.C.

Проверил: Преподаватель

\_\_\_\_\_\_\_\_\_\_\_Михайлов Д.В.

Великий Новгород

2021

1. **Формулировка цели и задач**

Рассмотреть представление знаний фреймами и соответствующие способы управления выводом.

1. **Модель представления знаний**

Предметная область мобильные устройства. Их я разбил на телефоны и планшеты.

Фреймом называется структура для описания понятия или ситуации, состоящая из характеристик этой ситуации и их значений.

Особенность фреймовых моделей представления знаний состоит в том, что все понятия, описываемые в каждом из узлов модели, определяются набором атрибутов и их значениями, которые содержатся в слотах фрейма.

Слот — это атрибут, который связывается с узлом в системе, основанной на фреймах. Он является основной составляющей фрейма. Он имеет имя слота и значение слота.

С каждым слотом может быть связана одна или несколько процедур, которые выполняются, когда изменяются значения слотов. Чаще всего со слотами связываются процедуры:

ЕСЛИ-ДОБАВЛЕНО (IF-ADDED) — выполняется в том случае, когда новая информация помещается в слот;

ЕСЛИ-УДАЛЕНО (IF-REMOVED) — выполняется при удалении информации из слота;

ЕСЛИ-НУЖНО (IF-NEEDED) — выполняется при запросе информация из слота в том случае, когда слот пуст.

В моей работе реализованы только процедуры IF-ADDED.

Эти процедуры следят за изменением информации, приписываемой данному узлу, и проверяют, что при изменении тех или иных значений производятся соответствующие действия.

Фреймовые системы позволяют использовать объектно-ориентированный подход к представлению знаний.

Фрейм «Мобильное устройство»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота | Демон |
| ОС | U(Unique) | Text | “Android”, “IOS” |  |
| Диагональ экрана | U(Unique) | Float |  | Проверка 0<= диагональ<=20 |

Фрейм «Телефон»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота | Демон |
| IS-A | U(Unique) | FRAME | Мобильное устройство |  |
| ОС | S(Same) | Text | “Android”, “IOS” |  |
| Диагональ экрана | S(Same) | Float |  | Проверка 0<= диагональ<=20 |
| Фронтальная камера | U(Unique) | Bool |  |  |
| 4G LTE | U(Unique) | Bool |  |  |
| Вспышка | U(Unique) | Bool |  |  |
| Внутренняя память | U(Unique) | Integer |  |  |
| 2 SIM карты | U(Unique) | Bool |  |  |

Фрейм «Планшет»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота | Демон |
| IS-A | U(Unique) | FRAME | Мобильное устройство |  |
| ОС | S(Same) | Text | “Android”, “IOS” |  |
| Диагональ экрана | S(Same) | Float |  | Проверка 0<= диагональ<=20 |
| Поддержка SIM карты | U(Unique) | Bool |  |  |
| Поддержка стилуса | U(Unique) | Bool |  |  |
| Поддержка клавиатуры | U(Unique) | Bool |  |  |

Фрейм «Телефон Iphone XS»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота | Демон |
| IS-A | U(Unique) | FRAME | Телефон |  |
| ОС | U(Unique) | TEXT | IOS |  |
| Диагональ экрана | U(Unique) | TEXT | 6 |  |
| Фронтальная камера | U(Unique) | BOOL | Да |  |
| 4G LTE | U(Unique) | BOOL | Да |  |
| Вспышка | U(Unique) | BOOL | Да |  |
| Внутренняя память | U(Unique) | INTEGER | 64 | Проверка Размер хранилища > 0 |
| 2 SIM карты | U(Unique) | BOOL | Да |  |

Фрейм «Планшет Galaxy Tab S7»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота | Демон |
| IS-A | U(Unique) | FRAME | Планшет |  |
| ОС | U(Unique) | Text | Android |  |
| Диагональ экрана | U(Unique) | Float | 11 | Проверка 0<= диагональ<=20 |
| Поддержка SIM карты | U(Unique) | Bool | Да |  |
| Поддержка стилуса | U(Unique) | Bool | Да |  |
| Поддержка клавиатуры | U(Unique) | Bool | Нет |  |

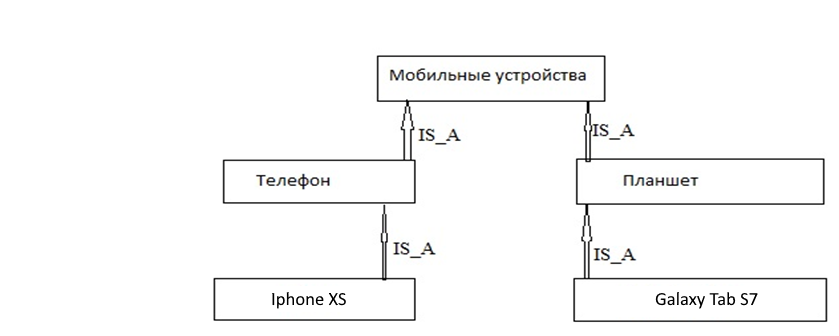
1. **Механизм управления вводом**

Фреймовые системы позволяют использовать объектно-ориентированный подход к представлению знаний. Эта структура базируется на отношениях IS-А между фреймами, описывающими некоторое мобильное устройство.

При этом наши представления и знания об объекте "Мобильное устройство" детализируем рядом понятий, которые определим через набор атрибутов и возможными типы их значений. Все атрибуты разместим в слотах фрейма.

Таким образом, на самом верхнем уровне иерархии определен фрейм “Мобильное устройство”. В данном случае мобильные устройства разделяются на планшеты и телефоны, которые определены в качестве соответствующих дочерних фреймов.

Фреймы, описывающие различные объекты называются шаблонами, а фреймы верхнего уровня, использующиеся для представления этих шаблонов, называются фреймами класса.



1. **Описание механизма поиска по запросу в поисковой системе**

В этой лабораторной работе предусмотрен поиск информации об определенной модели, мобильного устройства, это происходит путем ввода необходимой модели в окошко ввода, и программа предоставляет основную информацию об этой модели.

1. **Тестовые наборы для всех функций преобразования над знаниями в системе**

**На вход подается название модели мобильного устройства:**

Вводим: Iphone XS

Ответ:

название: Iphone XS

ОС: IOS

Диагональ экрана(дюймы): 6

Есть селфи камера: да

Есть поддержка 4G: да

Есть вспышка: да

Размер хранилища(ГБ): 64

Есть поддержка второй SIM карты: да

**Неудачный набор данных:**

Является ситуация при которой мы введем модель мобильного устройства, которой нет в базе:

Вводим: Realme 7

Ответ: «База знаний не содержит сведений о данной модели»

**Также при добавлении новой модели если она уже есть в базе:**

Вводим: Iphone XS

Ответ: «Данная модель уже существует в базе знаний»

**Демонстрация работы демона-процедуры IF-ADDED:**

Выберите действие(найти, добавить, удалить): добавить

Тип устройства(телефон, планшет): телефон

название: Iphone 13

ОС: IOS

Диагональ экрана(дюймы): 128

Есть селфи камера: да

Есть поддержка 4G: да

Есть вспышка: да

Размер хранилища(ГБ): -1

Есть поддержка второй SIM карты: нет

Ответ: Размер хранилища не может быть <= 0

**Вывод**

Были приобретены умения и навыки реализации пополняемой динамической базы знаний, основанной на фреймовой системе представления знаний. И также был разработан редактор фреймов, с помощью которого можно добавлять и удалять фреймы.

**Приложение:**

**Код программы**

import inspect

from abc import ABC

def dbool(bool\_str):

return bool\_str == 'True'

class MobileDevice(ABC):

def \_\_init\_\_(self, name, OS, display\_size):

self.\_\_name = name

self.OS = OS

self.display\_size = int(display\_size)

def get\_type(self):

raise NotImplementedError(

"Пожалуйста не используйте экземпляры базового класса")

def \_\_str\_\_(self):

return ','.join([self.get\_type(), \*map(str, self.\_\_dict\_\_.values())])

def deserialize(device\_str):

str\_parts = device\_str.split(',')

if str\_parts[0] == 'P':

return Phone(str\_parts[1], str\_parts[2], int(str\_parts[3]), dbool(str\_parts[4]), dbool(str\_parts[5]), dbool(str\_parts[6]), int(str\_parts[7]), dbool(str\_parts[8]))

else:

return Tablet(str\_parts[1], str\_parts[2], int(str\_parts[3]), dbool(str\_parts[4]), dbool(str\_parts[5]), dbool(str\_parts[6]))

@property

def name(self):

return self.\_\_name

@property

def OS(self):

return self.\_\_OS

@OS.setter

def OS(self, OS):

OS\_LIST = ['Android', 'IOS']

if OS in OS\_LIST:

self.\_\_OS = OS

else:

raise ValueError('Только Android и IOS ОС поддерживаются')

@property

def display\_size(self):

return self.\_\_display\_size

@display\_size.setter

def display\_size(self, display\_size):

if display\_size <= 0:

raise ValueError("Диагональ экрана не может быть <= 0")

if display\_size > 20:

raise ValueError(

"Устройства с диагональю экрана > 20 не являются мобильными")

self.\_\_display\_size = display\_size

class Phone(MobileDevice):

def \_\_init\_\_(self, name, OS, display\_size, selfie\_cam, LTE, flashlight, storage\_size, second\_SIM):

super().\_\_init\_\_(name, OS, display\_size)

self.\_\_selfie\_cam = selfie\_cam

self.\_\_LTE = LTE

self.\_\_flashlight = flashlight

self.storage\_size = int(storage\_size)

self.\_\_second\_SIM = second\_SIM

def get\_type(self):

return "P"

@ property

def storage\_size(self):

return self.\_\_storage\_size

@ storage\_size.setter

def storage\_size(self, storage\_size):

if storage\_size <= 0:

raise ValueError("Размер хранилища не может быть <= 0")

self.\_\_storage\_size = storage\_size

@ property

def selfie\_cam(self):

return self.\_\_selfie\_cam

@ property

def LTE(self):

return self.\_\_LTE

@ property

def flashlight(self):

return self.\_\_flashlight

@ property

def second\_SIM(self):

return self.\_\_second\_SIM

class Tablet(MobileDevice):

def \_\_init\_\_(self, name, OS, display\_size, SIM\_support, stylus\_support, keyboard\_support):

super().\_\_init\_\_(name, OS, display\_size)

self.\_\_SIM\_support = SIM\_support

self.\_\_stylus\_support = stylus\_support

self.\_\_keyboard\_support = keyboard\_support

def get\_type(self):

return "T"

@ property

def SIM\_support(self):

return self.\_\_SIM\_support

@ property

def stylus\_support(self):

return self.\_\_stylus\_support

@ property

def keyboard\_support(self):

return self.\_\_keyboard\_support

def parse\_db():

db = []

with open('db.txt') as db\_file:

for line in db\_file.readlines():

if len(line) > 0:

db.append(MobileDevice.deserialize(line.strip("\r\n ")))

return db

def save\_db():

with open('db.txt', 'w') as db\_file:

db\_file.write('\n'.join(map(str, db)))

def add\_entry(e):

if find\_entry(e.name) is None:

db.append(e)

save\_db()

else:

raise ValueError(f"Entry with same name is in DB already")

def find\_entry(name):

for e in db:

if e.name == name:

return e

return None

def remove\_entry(name):

e = find\_entry(name)

if e is not None:

db.remove(e)

save\_db()

else:

raise ValueError(f"Entry with this name not found in DB")

db = parse\_db()

def bool\_to\_ru(e):

if type(e) == bool:

return 'да' if e else 'нет'

return e

while True:

command = None

en\_to\_ru = {

"name": "название",

"OS": "ОС",

"display\_size": "Диагональ экрана(дюймы)",

"selfie\_cam": "Есть селфи камера",

"LTE": "Есть поддержка 4G",

"flashlight": "Есть вспышка",

"storage\_size": "Размер хранилища(ГБ)",

"second\_SIM": "Есть поддержка второй SIM карты",

"SIM\_support": "Есть поддержка SIM карты",

"stylus\_support": "Есть поддержка ситлуса",

"keyboard\_support": "Есть поддержка клавиатуры"

}

while command not in ['найти', 'добавить', 'удалить']:

command = input("Выберите действие(найти, добавить, удалить): ")

if command == 'найти':

e = find\_entry(input('название: '))

if e is not None:

for key in e.\_\_dict\_\_:

print(

f"{en\_to\_ru[key.split('\_\_')[1]]}: {bool\_to\_ru(e.\_\_dict\_\_[key])}")

else:

print(f"База знаний не содержит сведений о данной модели")

elif command == 'добавить':

try:

dclass = None

while dclass not in ['телефон', 'планшет']:

dclass = input("Тип устройства(телефон, планшет): ").lower()

if dclass == "телефон":

values = []

for key in list(inspect.signature(Phone.\_\_init\_\_).parameters)[1:]:

values.append(input(f"{en\_to\_ru[key]}: "))

for i in range(len(values)):

if values[i].lower() == 'да':

values[i] = True

elif values[i].lower() == 'нет':

values[i] = False

add\_entry(Phone(\*values))

else:

values = []

for key in list(inspect.signature(Tablet.\_\_init\_\_).parameters)[1:]:

values.append(input(f"{en\_to\_ru[key]}: "))

for i in range(len(values)):

if values[i].lower() == 'да':

values[i] = True

elif values[i].lower() == 'нет':

values[i] = False

add\_entry(Tablet(\*values))

except ValueError as e:

print(e)

else:

try:

remove\_entry(input('название: '))

except ValueError as e:

print(e)

**Содержимое db.txt**

P,Iphone XS,IOS,6,True,True,True,64,True

T,Galaxy Tab S7,Android,11,True,True,True