МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. Н. Э. Баумана

КАФЕДРА ПРОЕКТИРОВАНИЕ И ТЕХНОЛОГИЯ ПРОИЗВОДСТВА ЭЛЕКТРОННОЙ АППАРАТУРЫ

Отчет о выполнении

практического задания №1

«Теоретико-множественные операции»

по курсу «Функциональная логика и теория алгоритмов»

Выполнили: студенты гр. ИУ4-21Б

Горохов И.А.

Проверил: д.т.н., профессор, профессор каф. ИУ-4

Л.А. Зинченко

Москва 2023

Цель работы:

Разработка программы, выполняющей ограниченный набор действий над множествами.

1. Исходные данные:

Написать на языке С/C++/Python и отладить программу, реализующую следующие функции:

* Задание 2-х множеств;
* Ввод элементов множества;
* Вывод элементов множества;
* Добавление элемента множества;
* Удаление элемента множества.

2. Выполнение.

Программа вызывается с набором аргументов командной строки, включающих в себя команды и данные (аргументы команд). Команды выполняются последовательно, и в случае их успешного выполнения программа сохраняет состояние данных на момент завершения выполнения. В случае возникновения ошибки состояние хранимых данных не меняется (остается таким же, каким было в момент перед вызовом программы).

В качестве способа запросов выбран синтаксис схожий с утилитой ffmpeg, т.к. так же выполняются операции исключительно с пользовательскими данными:

-i name [element]... input set of elements and assign to name "name"   
-o name output set named "name"  
-a name element insert "element" into set named "name"  
-d name element delete "element" from set named "name"  
-L, -l list names of all sets  
-h help

В случае нарушения некорректного ввода (первый аргумент командной строки – не команда, или не подходящее количество аргументов). Данные о множествах хранятся в файле sets.json.

Содержимое основного кода программы находится в файле 1.py:

import sys, json

try:

with open("sets.json", "rt") as f:

sets = json.load(f)

except:

sets = dict()

def on\_error(description, task=None):

print(f"Error: {description}")

if task: print(f"while executing \"{ ' '.join(task) }\"")

print("Changes will not be applied. Use -h for help")

exit()

def help\_(\*args):

print("""usage: 1.py [{command} [argument]...]...

commands:

-i name [element]... input set of elements and assign to name "name"

-o name output set named "name"

-a name element insert "element" into set named "name"

-d name element delete "element" from set named "name"

-L, -l list names of all sets

-h help

""")

def list\_sets():

print(\*(sets.keys()))

def input\_set(name, \*elms):

if name in sets: print(f"set \"{name}\" exists and will be overwriten")

sets[name] = []

for i in elms: add\_to\_set(name, i)

print(f"\"{name}\" created")

def output\_set(name):

if sets[name]:

set\_str = "{"+(', '.join(sets[name]))+"}"

else:

set\_str = "\u2205"

print(f"{name}={set\_str}")

def add\_to\_set(name, elm):

if elm not in sets[name]:

sets[name].append(elm)

def del\_from\_set(name, elm):

try:

sets[name].remove(elm)

except ValueError:

print(f"NIL: {elm} not in set {name}")

actions = {

"-i": input\_set,

"-o": output\_set,

"-a": add\_to\_set,

"-d": del\_from\_set,

"-h": help\_,

"-L": list\_sets,

"-l": list\_sets,

None: lambda x:None, # for triggering task execution before exit

}

def parse(args):

task = [args[0]]

for arg in args[1:]:

if arg in actions:

try:

actions[task[0]](\*task[1:])

except TypeError:

on\_error("wrong amount of arguments", task)

except KeyError:

on\_error("no set with such name", task)

except ValueError:

on\_error("no set with such name", task)

task.clear()

task.append(arg)

return

if \_\_name\_\_ == "\_\_main\_\_":

args = list(sys.argv[1:])+[None]

if len(args)<=1:

print("No arguments provided. Use -h for help")

elif args[0] not in actions:

on\_error(f"first argument \"{args[0]}\" is not valid command")

parse(args)

if sets:

with open("sets.json", "wt") as f:

json.dump(sets, f)

3. Результаты работы.

Пример вызовов программы находится в файле 1.bat:

echo usage example

1.py -h -i set1 1 2 3 -o set1 -d set1 3 -a set1 stul -o set1 -l

1.py -o set123456789

1.py -d set1 x

pause

Результат её работы:

C:\Users\Иван\Documents\FLiTA>echo usage example

usage example

C:\Users\Иван\Documents\FLiTA>1.py -h -i set1 1 2 3 -o set1 -d set1 3 -a set1 stul -o set1 -l

usage: 1.py [{command} [argument]...]...

commands:

-i name [element]... input set of elements and assign to name "name"

-o name output set named "name"

-a name element insert "element" into set named "name"

-d name element delete "element" from set named "name"

-L, -l list names of all sets

-h help

set "set1" exists and will be overwriten

"set1" created

set1={1, 2, 3}

set1={1, 2, stul}

set1 8 2

C:\Users\Иван\Documents\FLiTA>1.py -o set123456789

Error: no set with such name

while executing "-o set123456789"

Changes will not be applied. Use -h for help

C:\Users\Иван\Documents\FLiTA>1.py -d set1 x

NIL: x not in set set1

C:\Users\Иван\Documents\FLiTA>pause

Для продолжения нажмите любую клавишу . . .

4. Вывод:

Программа корректно исполняет поставленные задачи, а именно:

* Задание двух множеств (или произвольного количества множеств);
* Ввод элементов множества;
* Вывод элементов множества;
* Добавление элемента множества;
* Удаление элемента множества.