МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

## Факультет информационных технологий и робототехники

Кафедра программного обеспечения информационных систем и технологий

**Отчет по лабораторной работе № 3**

по дисциплине:” Системное программирование”

на тему: “Командный язык и скрипты Shell”

Выполнил**:** студент группы 10702121

Меркулова М. С.

Принял**:** Давыденко Н.В.

Минск 2023

# Лабораторная работа № 4. Командный язык и скрипты Shell

Цель работы: Закрепить на практике принципы создания проектов с помощью скрип- тов SHELL, освоить средства примитивного графического интерфейса в Linux-скриптах.

**Задание 1**

1. Сделать пример с двумя вложенными диалогами типа YesNo.
2. Сделать пример, запрашивающий сначала имя человека, а потом

профессию. Вывести имя + профессию, прочитанные в диалоге.

1. Вывести **список** с названиями валют. После выбора валюты система

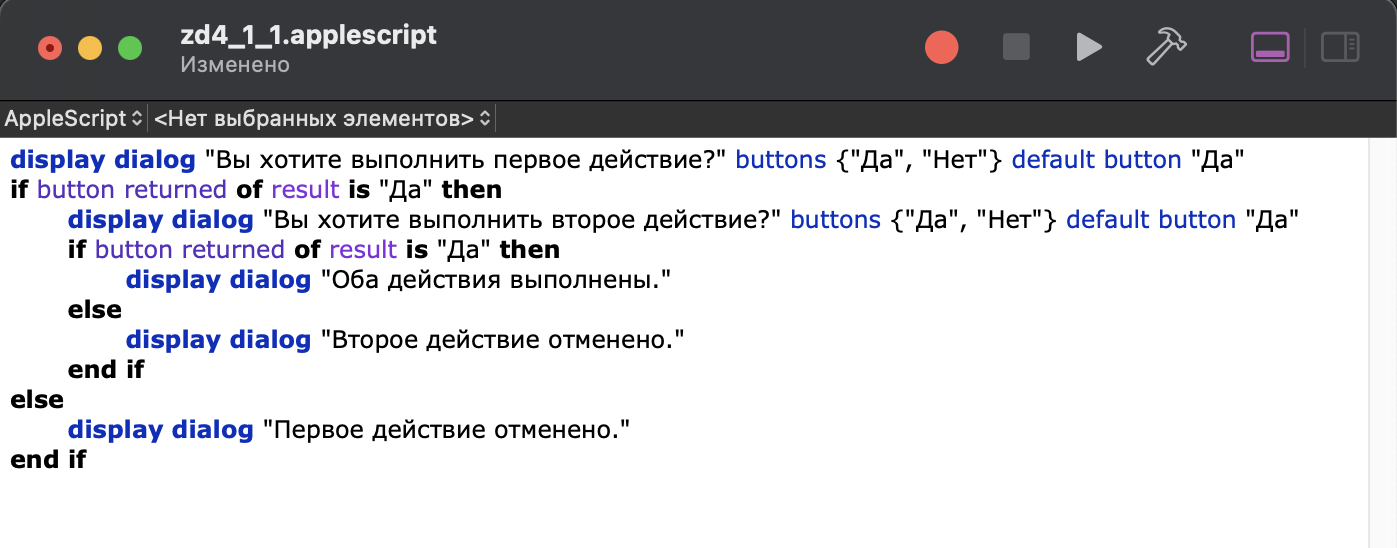
должна вывести ее котировку.

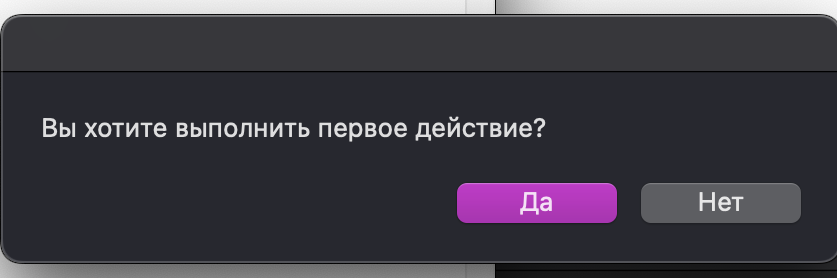
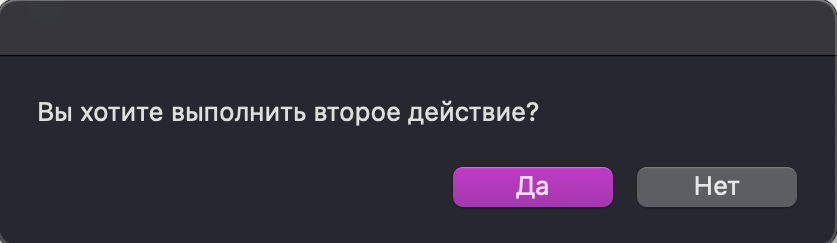
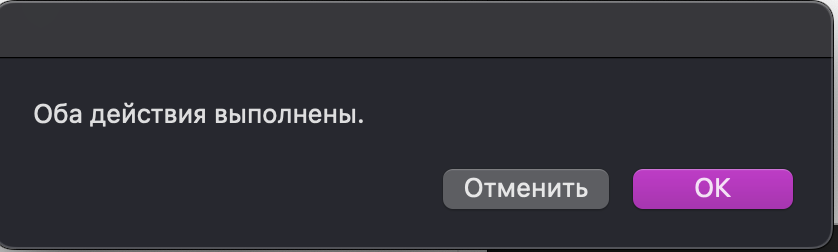
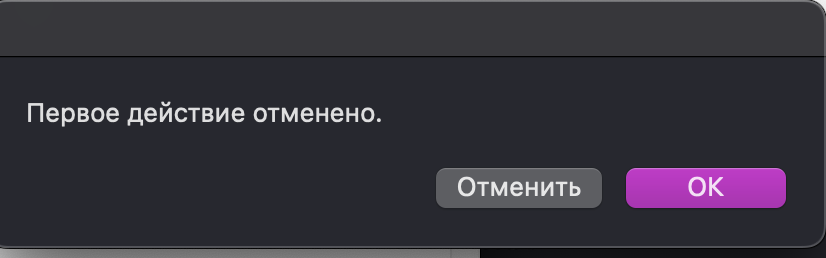
1. Измените предыдущую программу так, вместо списка валют предлагалось оконное **меню** валют. Чтобы программа работала в цикле. Для выхода из цикла нужно вместо названия валюты вводить exit.
2. Измените предыдущую программу так, для выхода из программы в меню была кнопка закрытия программы «exit».

**Решение**

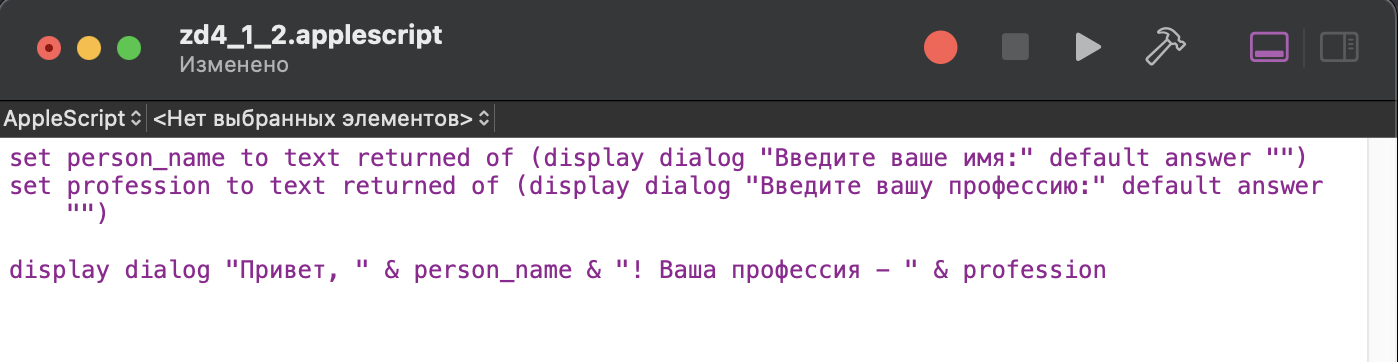
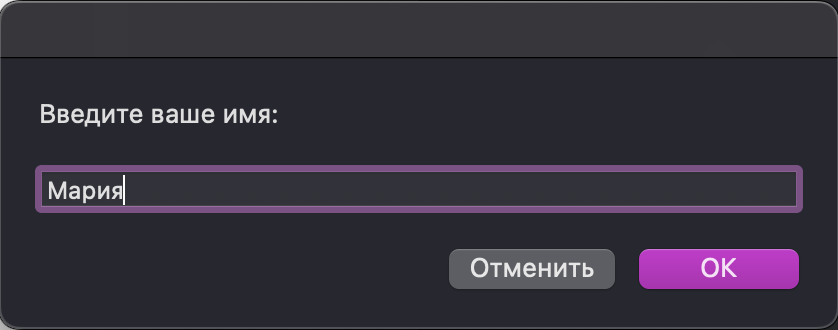
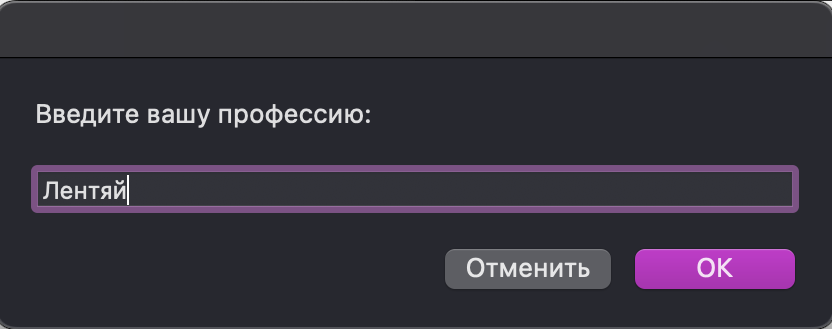
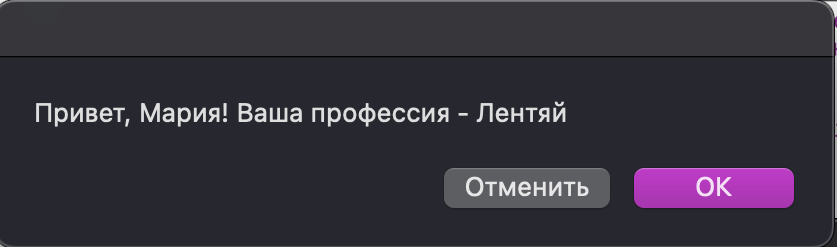
**Задание 1.1**

Создаем скрипт для решения

****

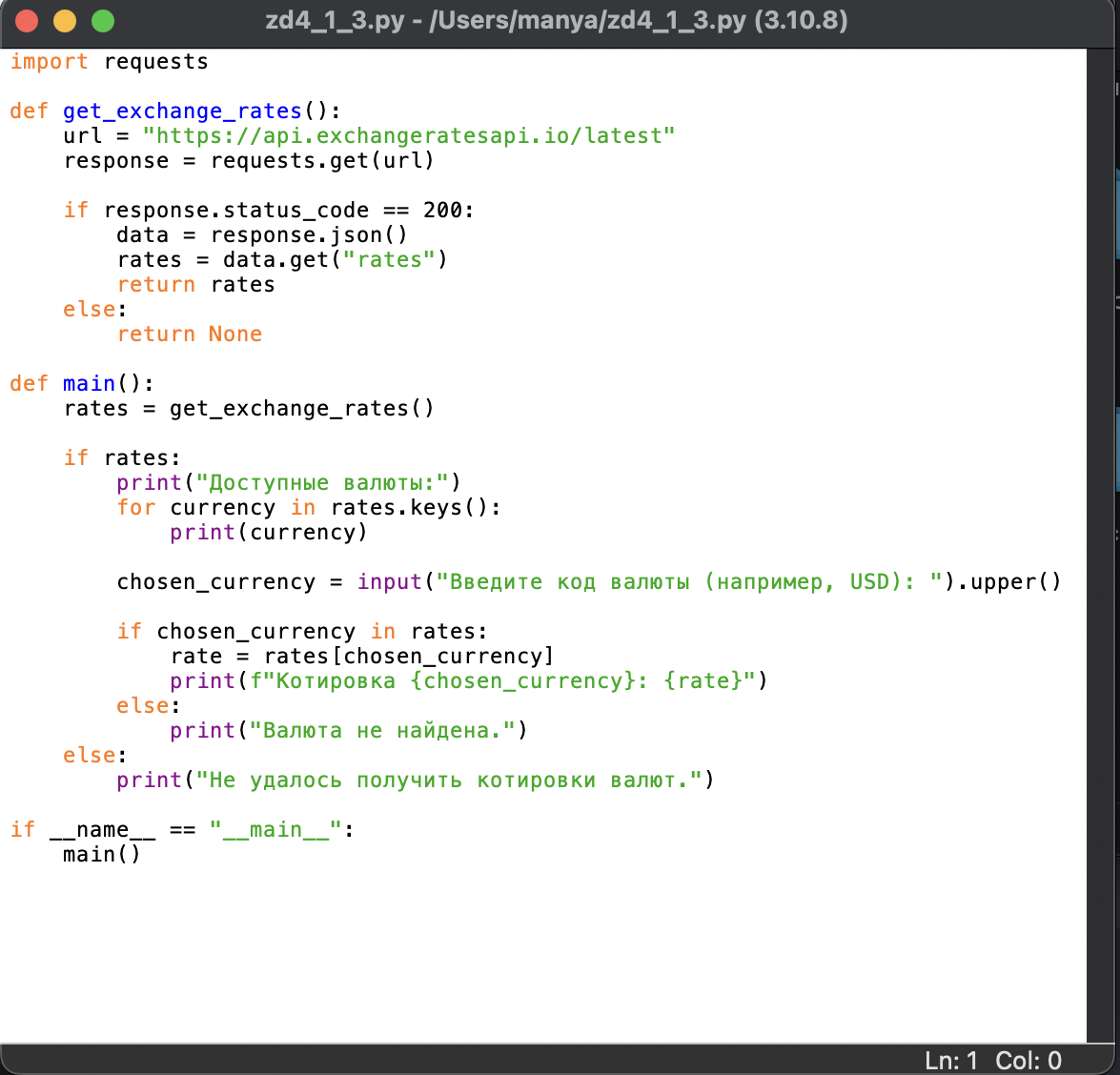
**** ****  

**Задание 1.2**

**** ****  

**Задание 1.3**

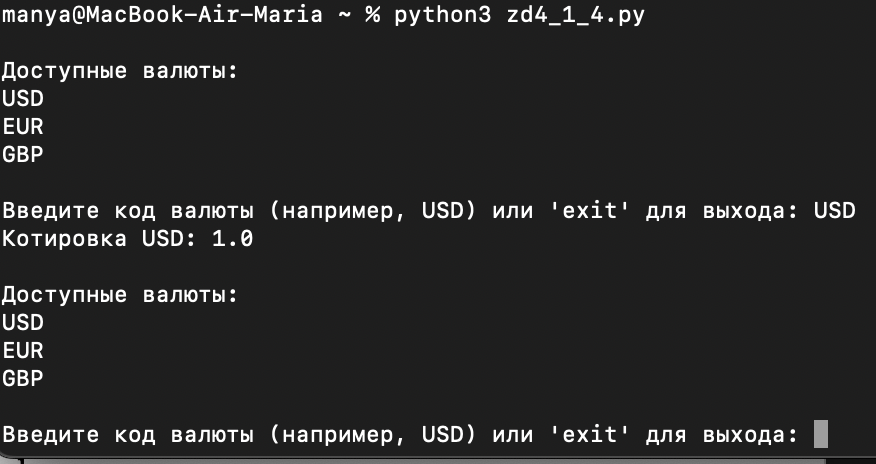
**Создадим скрипт на языке питон**

****

****

**Задание 1.4 и 1.5**

****

****

**Задание 2**

Создайте копию проекта представелнного в упражнении No3. Внесем изменеия в новом проекте, добавив вывод информации о домашнем каталоге каждого пользователя и включив в вывод общее число файлов и подкаталогов в каждом из них:

report\_home\_space () {  
local format="%8s%10s%10s\n"  
local i dir\_list total\_files total\_dirs total\_size user\_name if [[ $(id -u) -eq 0 ]]; then

dir\_list=/home/\*

user\_name="All Users" else

dir\_list=$HOME

user\_name=$USER fi

echo "<H2>Home Space Utilization ($user\_name)</H2>"

for i in $dir\_list; do total\_files=$(find $i -type f | wc -l)

total\_dirs=$(find $i -type d | wc -l) total\_size=$(du -sh $i | cut -f 1) echo "<H3>$i</H3>"  
echo "<PRE>"

printf "$format" "Dirs" "Files" "Size" printf "$format" "----" "-----" "----"

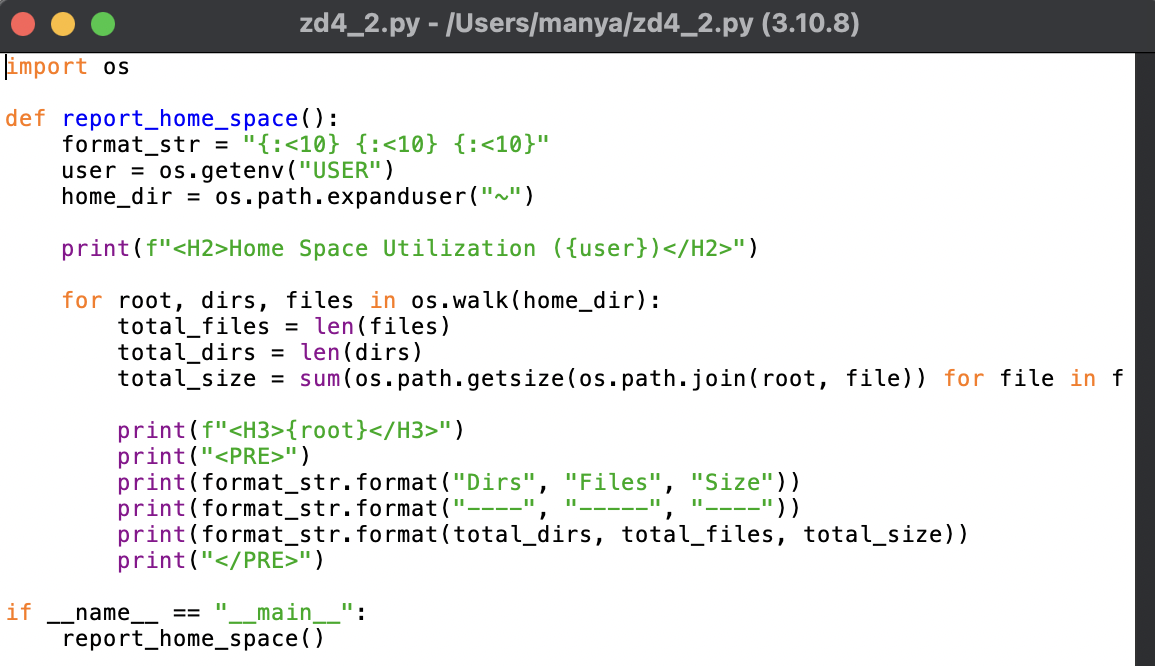
printf "$format" $total\_dirs $total\_files $total\_size

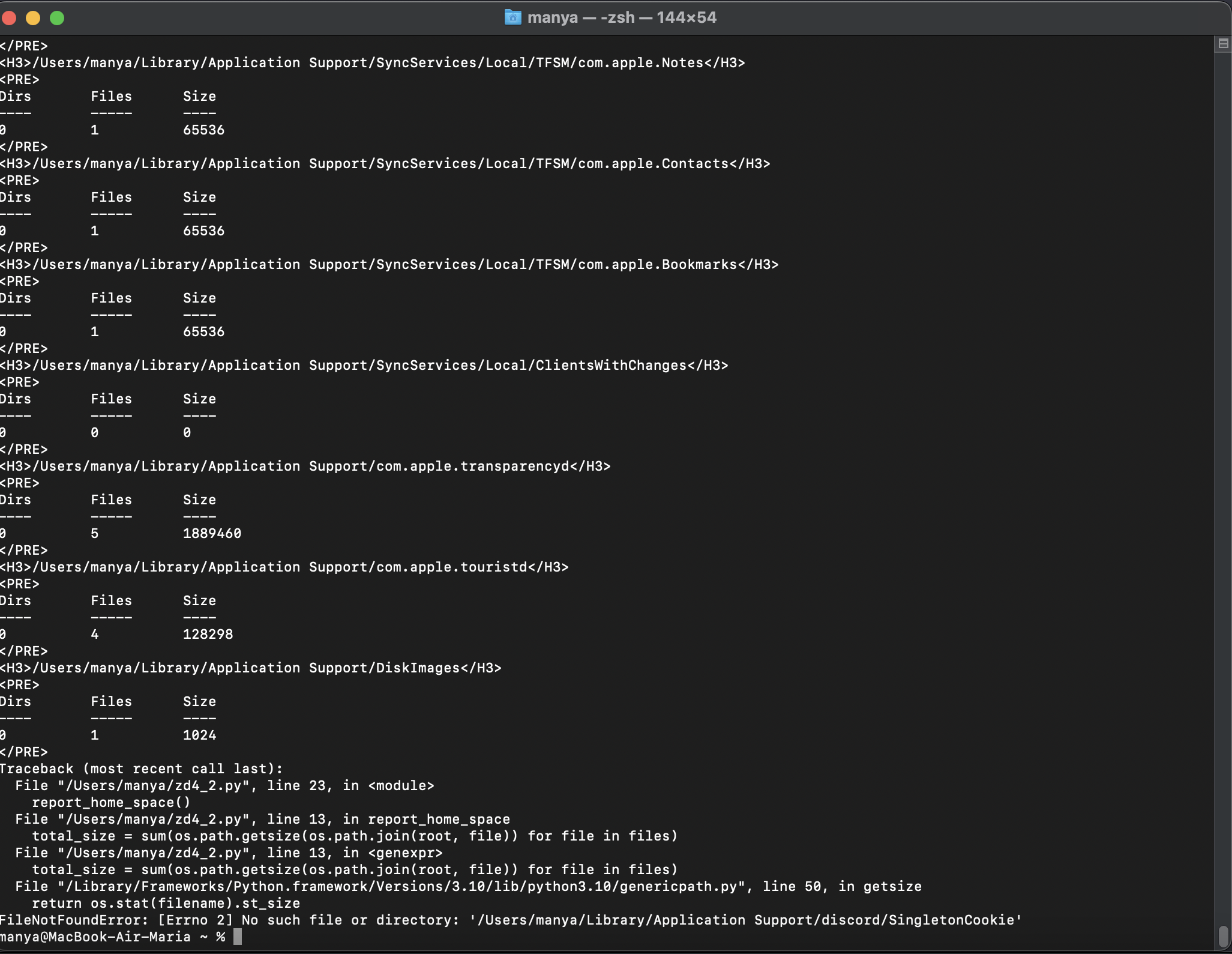
echo "</PRE>" done

return }

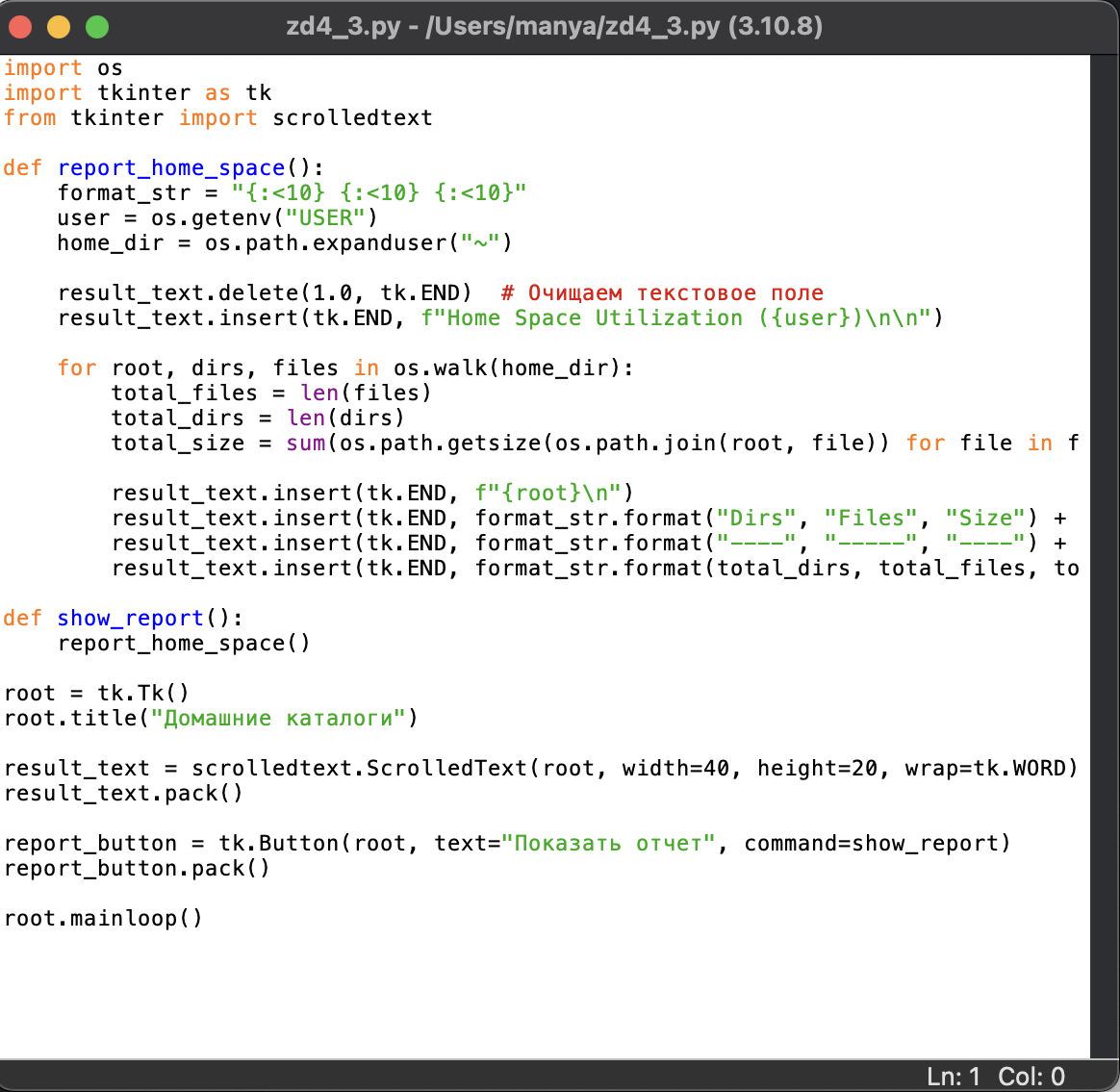
В этой новой версии проверяется наличие привилегий суперпользователя, но вместо того, чтобы выполнить полный набор операций в каждой из ветвей if, здесь устанавливаются некоторые переменные, которые затем используются в цикле for. В функции использованы несколько локальных переменных и команда printf для форматирования части вывода.

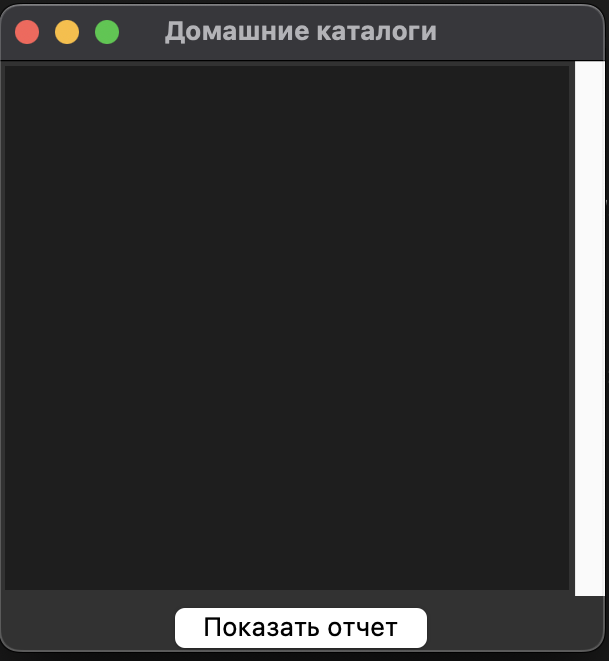
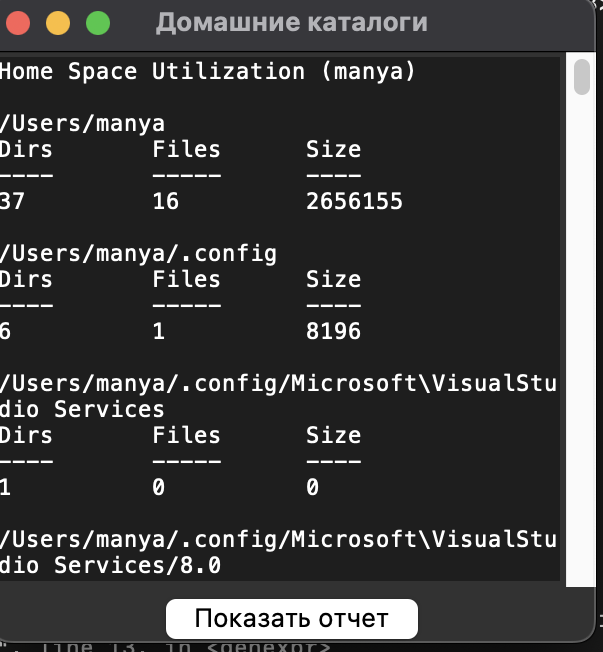
**Решение**

****

****

**Задание 3**

Для сценария выполненного в задании No2 создайте графическое диалоговое окно. **Решение** ****

**** ****

Контрольные вопросы:

1. Что такое *переменный* и *константы*, чем они отличаются? Как их различает командная оболочка?

Переменные и константы - это два различных концепта в программировании, и их отличия заключаются в их назначении и свойствах:

**Переменные:**

**• Изменяемые значения:** Переменные представляют собой именованные места в памяти, в которых можно хранить данные. Значения переменных могут изменяться в течение выполнения программы.

**• Пример:** В переменной **x** можно хранить число, и это число может изменяться в процессе выполнения программы.

**Константы:**

**• Незаменимые значения:** Константы представляют собой значения, которые остаются неизменными во время выполнения программы. Они обычно используются для представления фиксированных данных, таких как числа, строки или другие значения, которые не должны изменяться.

**• Пример:** Константа **PI** может быть установлена на значение 3.14159265 и оставаться неизменной во всех частях программы.

Командная оболочка, такая как Bash, также различает переменные и константы, но обычно не предоставляет явного механизма для определения констант. В Bash, переменные объявляются и используются, как вам угодно, и их значения могут изменяться. Пример объявления и использования переменной в Bash:

my\_variable="Hello, World!"

echo $my\_variable

Чтобы определить константы в Bash, обычно используются именованные переменные в верхнем регистре с ключевым словом **readonly**, чтобы предотвратить изменение их значений:

readonly PI=3.14159265

echo $PI

Это не сделает переменную **PI** полностью неизменной, но предотвратит ее изменение с помощью обычных операций присваивания. Тем не менее, важно понимать, что Bash не предоставляет строгой концепции констант, как многие другие языки программирования, и вам нужно будет соблюдать соглашения и дисциплину, чтобы сделать переменные "константами".

1. С помощью каких утилит реализуется средства примитивного графического интерфейса в Linux-скриптах?

В Linux-скриптах для создания примитивного графического интерфейса часто используются следующие утилиты:

**Dialog:** Утилита **dialog** предоставляет набор диалоговых окон, которые можно использовать для создания текстовых интерфейсов в терминале. Она позволяет создавать диалоговые окна для ввода текста, выбора опций и многое другое. Пример использования:

codedialog --msgbox "Привет, мир!" 10 30

**Whiptail:** **Whiptail** - это форк утилиты **dialog** с некоторыми улучшениями и дополнительными функциями. Она также позволяет создавать текстовые диалоговые окна в терминале. Пример использования:

codewhiptail --msgbox "Привет, мир!" 10 30

**YAD (Yet Another Dialog):** **YAD** - это более мощная утилита, которая позволяет создавать графические диалоговые окна с использованием GTK. Она предоставляет больше возможностей для создания интерактивных интерфейсов в скриптах. Пример использования:

codeyad --info --text "Привет, мир!" --title "Пример"

**Zenity:** **Zenity** - это еще одна утилита, которая использует GTK для создания графических диалоговых окон в терминале. Она предоставляет широкий набор виджетов для создания различных типов диалогов. Пример использования:

codezenity --info --text "Привет, мир!" --title "Пример"

Эти утилиты позволяют добавить интерактивность в ваши Linux-скрипты, создавая простые графические диалоговые окна, которые пользователи могут использовать для ввода данных или принятия решений. Выбор конкретной утилиты зависит от ваших потребностей и предпочтений.

Вывод*:* В ходе выполнения работы были закреплены на практике принципы создания проектов с помощью скрип- тов SHELL, освоены средства примтивного графического интерфейса в Linux-скриптах.