

Лабораторная работа 13

Средства, применяемые при
разработке программного обеспечения в ОС типа UNIX/LinuxПетрова

Мария Евгеньевна

Содержание

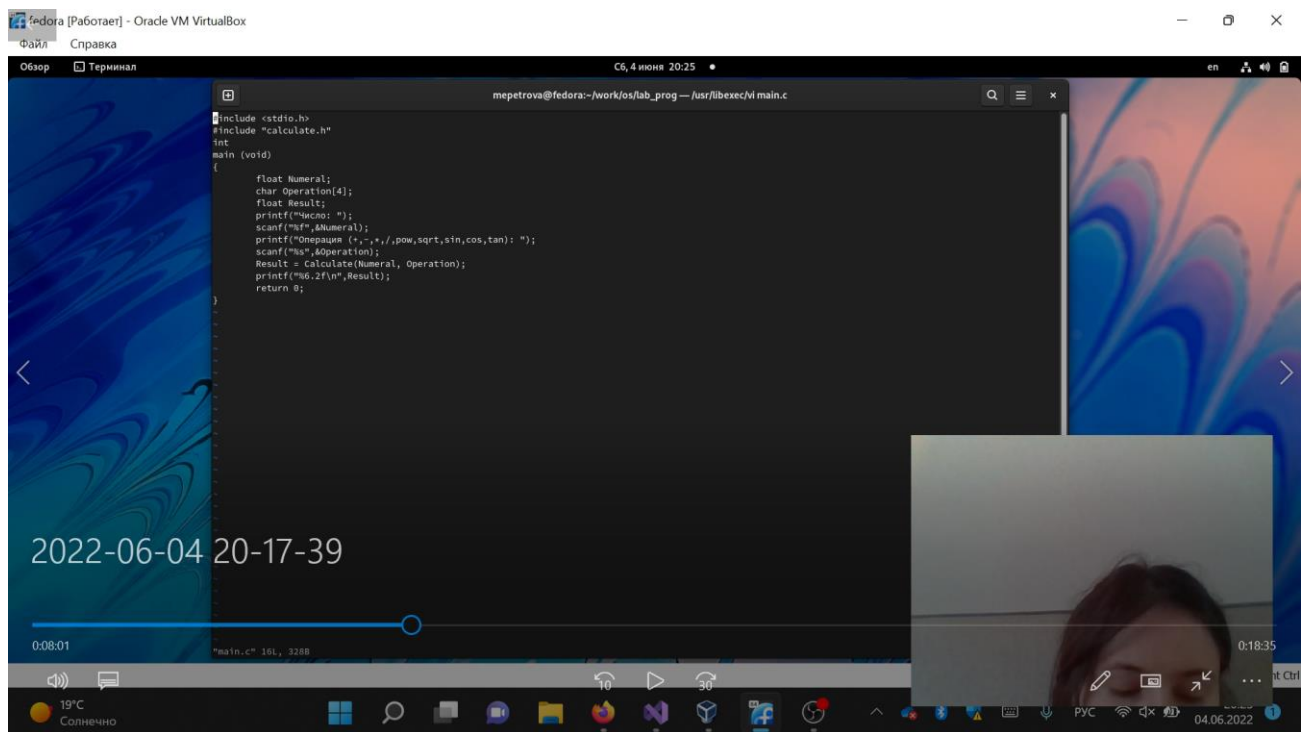
Цель работы	1
Выполнение лабораторной работы	1
Вывод.....	5
Контрольные вопросы	5
Контрольные вопросы.....	5

Цель работы

Приобрести простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями.

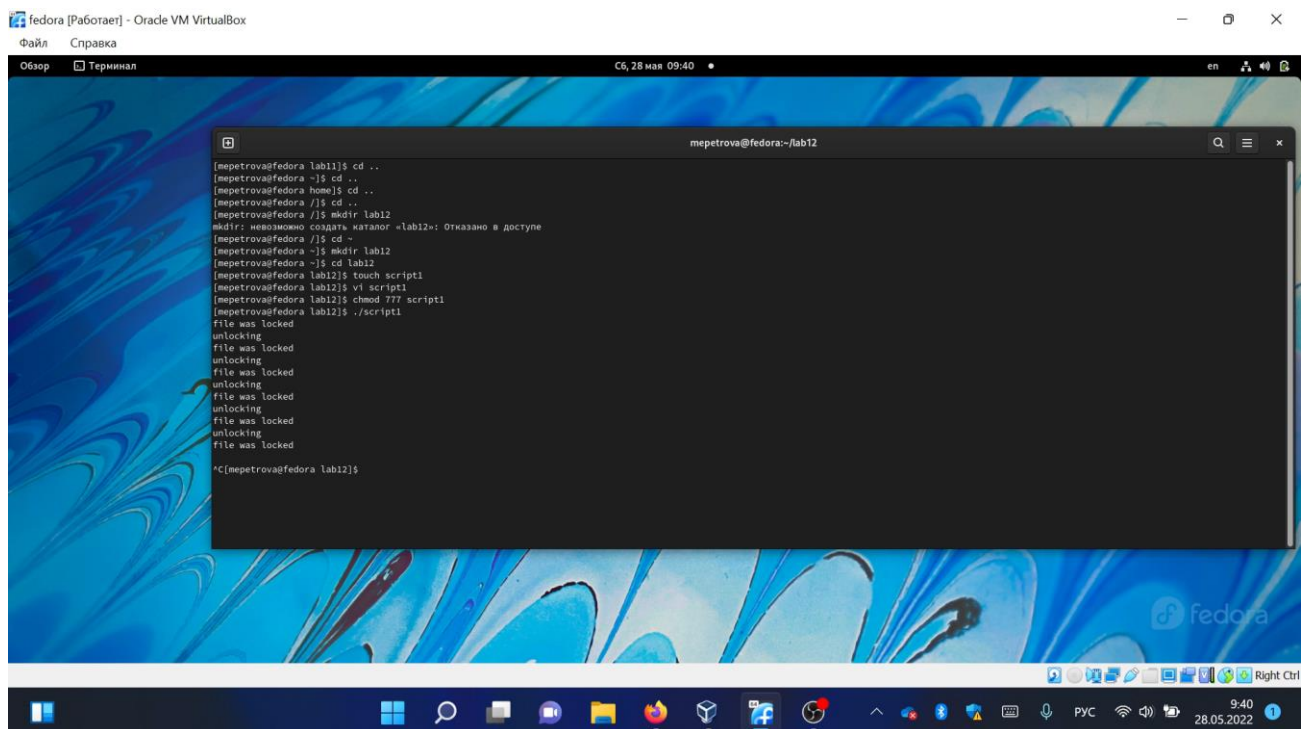
Выполнение лабораторной работы

1. Скрипт 1 (рис. 1)



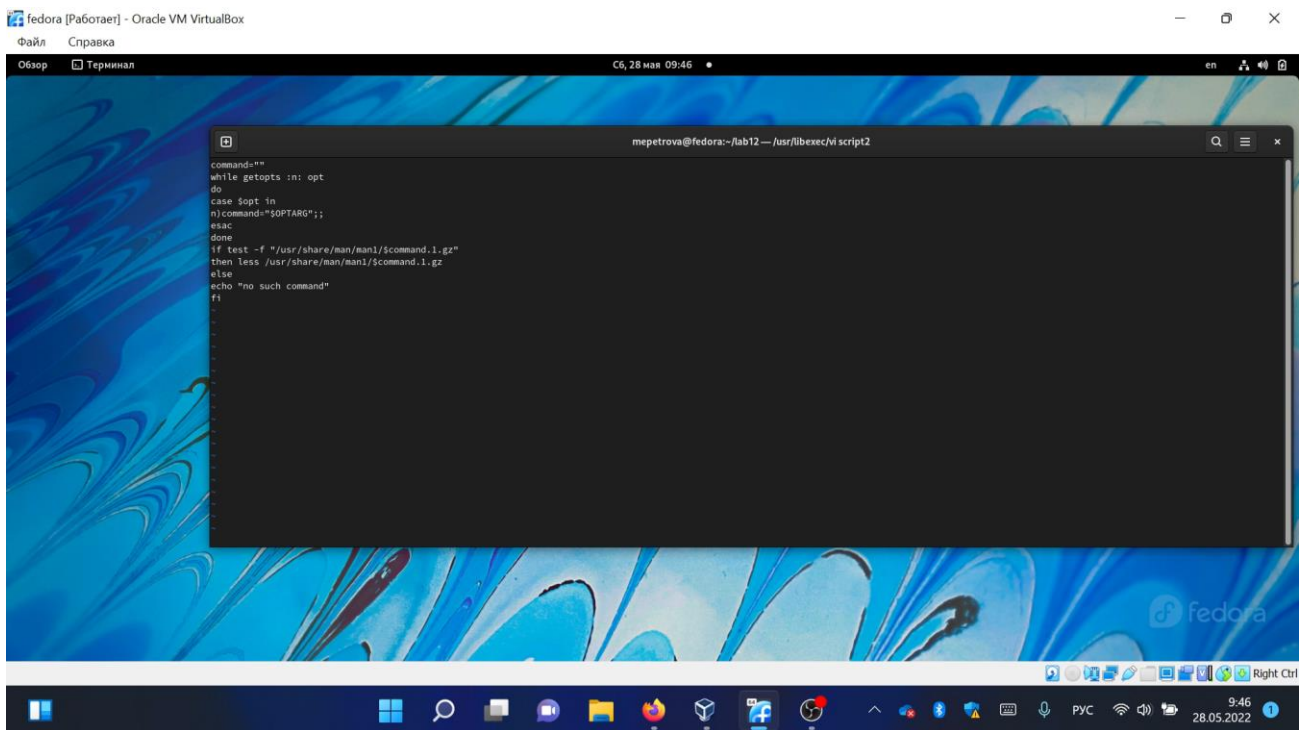
Скрипт 1

Запуск (рис. 2)



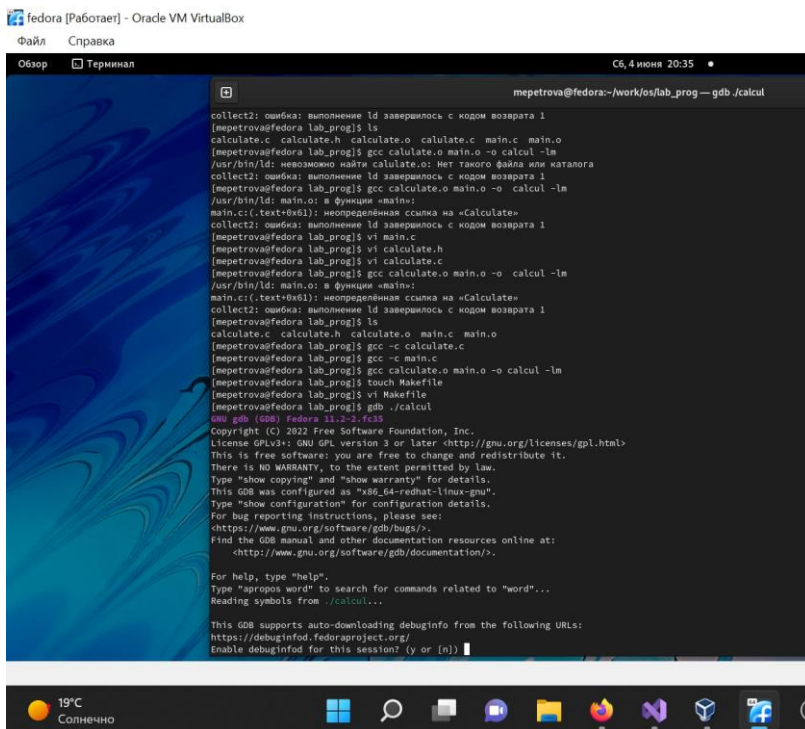
Скрипт 1 запуск

2. Скрипт 2 (рис. 3)



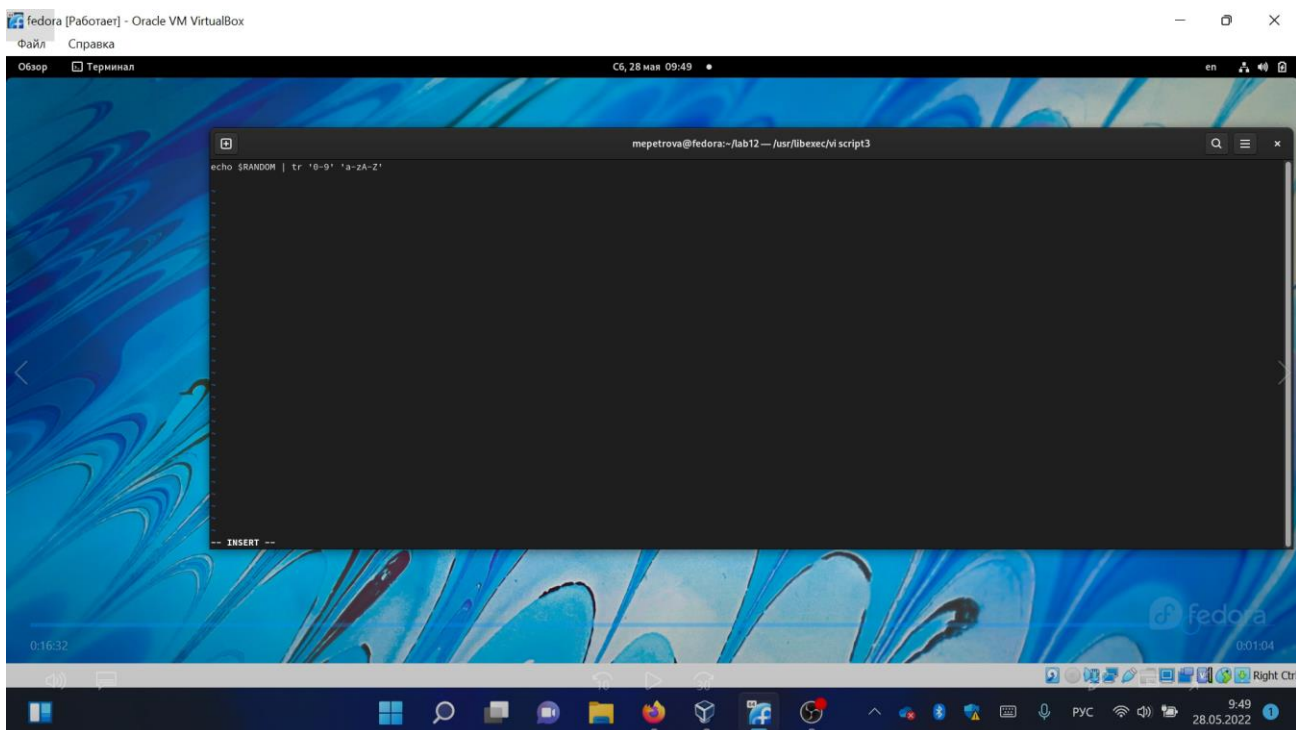
Скрипт 2

Запуск (рис. 4)



Скрипт 2 запуск

3. Скрипт 3 (рис. 5)



Скрипт 3

Запуск (рис. 6)

```

7
8 float Numeral;
9 char Operation[4];
10 float Result;
(gdb) list 12,15
12 scanf("%f",&Numeral);
13 printf("Numeral: %f\n",pow,sqrt,sin,cos,tan);
14 scanf("%s",Operation);
15 Result = Calculate(Numeral,Operation);
(gdb) list calculate.c:20,29
20 scanf("%f",&SecondNumeral);
21 return Numeral - SecondNumeral;
22 }
23 else if(strcmp(Operation,"*"),1) == 0)
24 {
25 printf("Numeral: %f\n");
26 scanf("%f",&SecondNumeral);
27 return Numeral * SecondNumeral;
28 }
29 else if(strcmp(Operation,"/"),1) == 0)
(gdb) list calculate.c:20,27
20 scanf("%f",&SecondNumeral);
21 return Numeral - SecondNumeral;
22 }
23 else if(strcmp(Operation,"*"),1) == 0)
24 {
25 printf("Numeral: %f\n");
26 scanf("%f",&SecondNumeral);
27 return Numeral * SecondNumeral;
(gdb) break 25
Breakpoint 1 at 0x401201: file calculate.c, line 25.
(gdb) info breakpoints
Num Type Disp Enb Address What
1 breakpoint keep y 0x0000000000401201 in calculate at calculate.c:25
(gdb) run
Starting program: /home/abkhoshkoev/work/os/lab_prog/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): *
Breakpoint 1, Calculate (Numeral=5, Operation=0x7fffffffdf14 "<
25 printf("Numeral: %f\n");
(gdb) backtrace
#0 Calculate (Numeral=5, Operation=0x7fffffffdf14 "<
#1 0x0000000000401201 in main () at main.c:15
(gdb) print Numeral
$1 = 5
(gdb) display Numeral

```

Скрипт 3 запуск

Вывод

Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Контрольные вопросы

Контрольные вопросы

1: Найдите синтаксическую ошибку в следующей строке: `while [$1 != "exit"]`

\$1. Так же между скобками должны быть пробелы. В противном случае скобки и рядом стоящие символы будут восприниматься как одно целое

2: Как объединить (конкатенация) несколько строк в одну?

```
cat file.txt | xargs | sed -e 's/\. /\n/g'
```

3: Найдите информацию об утилите `seq`. Какими иными способами можно реализовать её функционал при программировании на `bash`?

`seq` - выдает последовательность чисел. Реализовать ее функционал можно командой `for n in {1..5} do done`

4: Какой результат даст вычисление выражения `$((10/3))`?

3

5: Укажите кратко основные отличия командной оболочки `zsh` от `bash`.

`Zsh` очень сильно упрощает работу. Но существуют различия. Например, в `zsh` после `for` обязательно вставлять пробел, нумерация массивов в `zsh` начинается с 1 (что не особо удобно на самом деле). Если вы собираетесь писать скрипт, который легко будет запускать множество разработчиков, то я рекомендую `Bash`. Если скрипты вам не нужны - `Zsh` (более простая работа с файлами, например)

6: Проверьте, верен ли синтаксис данной конструкции `for ((a=1; a <= LIMIT; a++))`

Верен

7: Сравните язык `bash` с какими-либо языками программирования. Какие преимущества у `bash` по сравнению с ними? Какие недостатки?

`Bash` позволяет очень легко работать с файловой системой без лишних конструкций (в отличие от обычного языка программирования). Но относительно обычных языков программирования `bash` очень сжат. Тот же `C` имеет гораздо более широкие возможности для разработчика.