# **Лаба 1**

## **Задание 1**

### **mkdir** – создать директорию

**Ключи: -р** создавать родительские

mkdir -p lab0/dir1/dir2

### **echo** – вывести строку текст

**Ключи: -е** распознавание \n и других

echo -e “text\nline2”>filename

### **cat** – объединять файлы и печатать в стандартный вывод

**Ключи: -n** нумеровать строки

cat -n file1

### **touch** – создать файл

touch filename1 filename2 filename3

### **ls** – посмотреть что в папке

**Ключи: -a** все файлы, в том числе скрытые

**-с** сортировать по времени модификации или создания, начиная с новых

**-С** выводить колонками

**-d** выводить только директории

**-f** не сортировать

**-F** показывать тип объекта, добавление \*/=>@|

**-l** подробный список

**-L** к символическим ссылкам добавить куда они обращаются

**-Q** имена папок и файлов в кавычки

**-r** обратный порядок сортировки

**-R** рекурсивно отображать поддиректории

**-s** выводить размер файла в блоках

**-S** сортировать по размеру, сначала большие

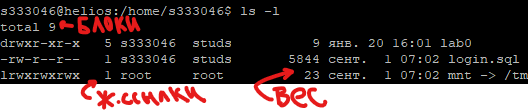
**-t** сортировать по времени последней модификации

**-u** сортировать по времени последнего доступа

**-U** не сортировать

**-X** сортировать по алфавиту

**-1** отображать один файл на одну строку



### **pwd** - вывести в терминал путь к текущей папке

### **cd** – поменять рабочую директорию

**cd** в домашнюю

**cd ~**  в домашнюю

**cd -**  в предыдущую

**cd /**  в корневую

**cd ./**  текущая

**cd ..** вверх

### **more** - просмотра файлов в терминале

### **cp** – копирование или переименование

cp oldname newname

cp что куда

### **rm** – удалить файл или директорию

**Ключи: -r** рекурсивно удалить всё

rm -r directoryname

### **rmdir** – удалить директорию

rmdir dirname

### **mv** – переместить или переименовать

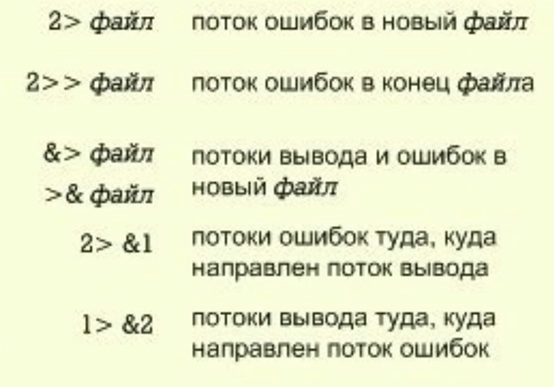
mv oldname newname

cp что куда

### **ПОТОКИ:**

Всего три потока:

* **stdin** — стандартный ввод (клавиатура) - **0**
* **stdout** — стандартный вывод (экран) - **1**
* **stderr** — стандартная ошибка (вывод ошибок на экран) - **2**



## **Задание 2**

### **chmod** – присвоить права

**Ключи**: **-R** рекурсивно

**a=** удалить все

**о=** удалить другим пользователям

**о-4** нельзя

**uo=rwx** юзер и другие пользователи все права

## **Задание 3**

### **cp** – копирование или переименование

**Ключи: -r** рекурсивно

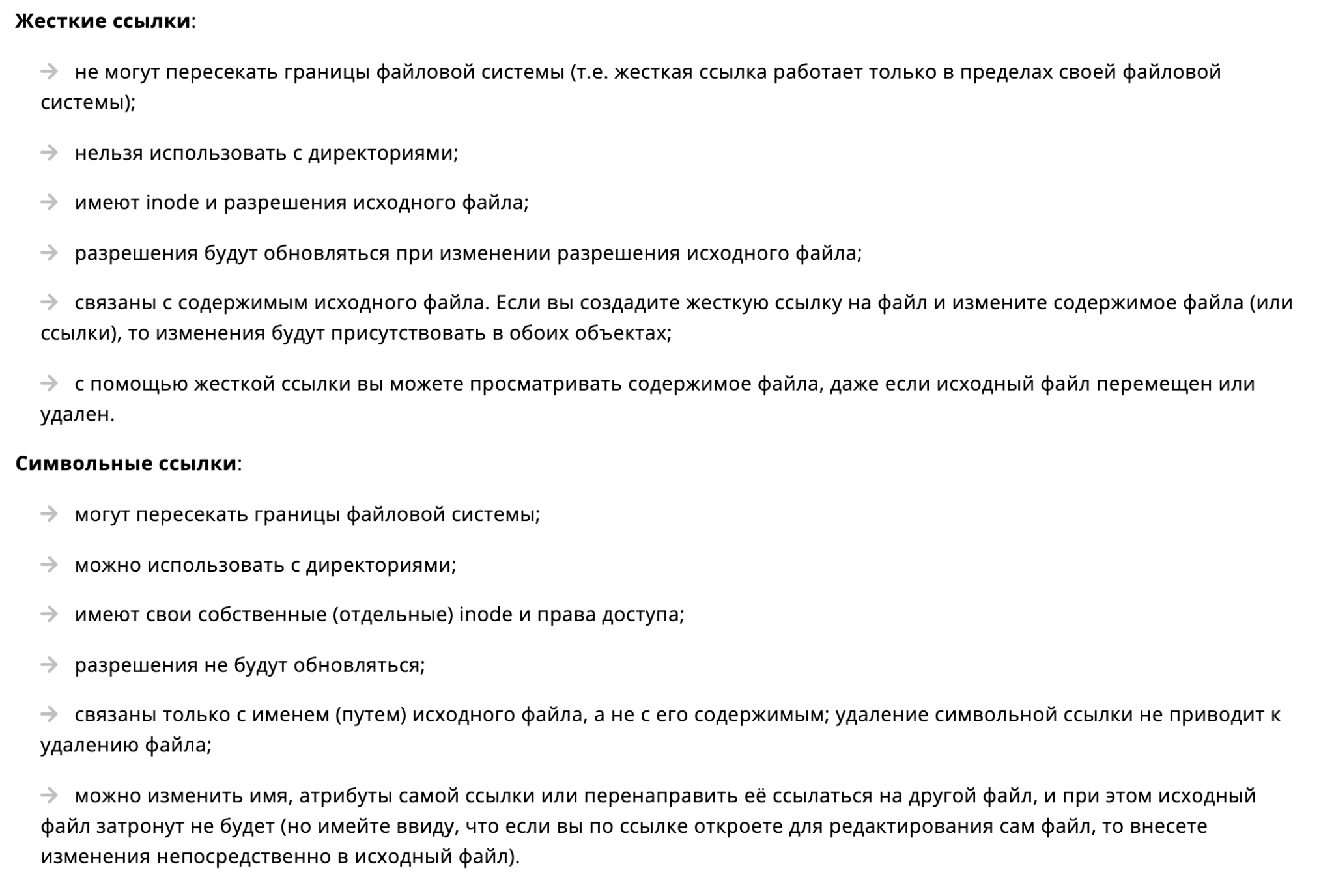
cp -r oldname newname

cp что куда

### **ln** – создать ссылку

**Ключи: -s** софт линк

ln путь к файлу от ссылки путь к ссылке с текущей позиции



### **cat** – тут объединить файлы

cat file1 file2 > newfilename

## **Задание 4**

### **cat** - вывести содержимое файла

cat ./\*r ./\*/\*r ./\*/\*/\*r рекурсивно вывести содержимое фай оканчивающихся на

### **wc** - подсчитать количество строк или слов в тексте

**Ключи: -с** размер объекта в байтах

**-m** количество символов

**-l** количество строк

**-w** количество слов

wc -l \*3 \*/\*3 подсчитать колво строк в файлах на \*3 в директории

### **head** - вывести сколько то с начала

cat file1 | head -3

### **tail** - вывести сколько то с конца

cat file1 | tail -3

### **echo** - вывести или перенаправить

### **sort** - сортировка

**Ключи: -b** не учитывать пробелы

**-d** использовать для сортировки только буквы и цифры

**-i** сортировать только по ASCII символах

**-n** сортировка строк linux по числовому значению

**-r** сортировать в обратном порядке

**-с** проверить был ли отсортирован файл

**-o** вывести результат в файл

**-u** игнорировать повторяющиеся строки

**-m** объединение ранее отсортированных файлов

**-k** указать поле по которому нужно сортировать строки, если не задано, сортировка выполняется по всей строке

**-f** использовать в качестве разделителя полей ваш символ вместо пробела

### **grep** - поиск

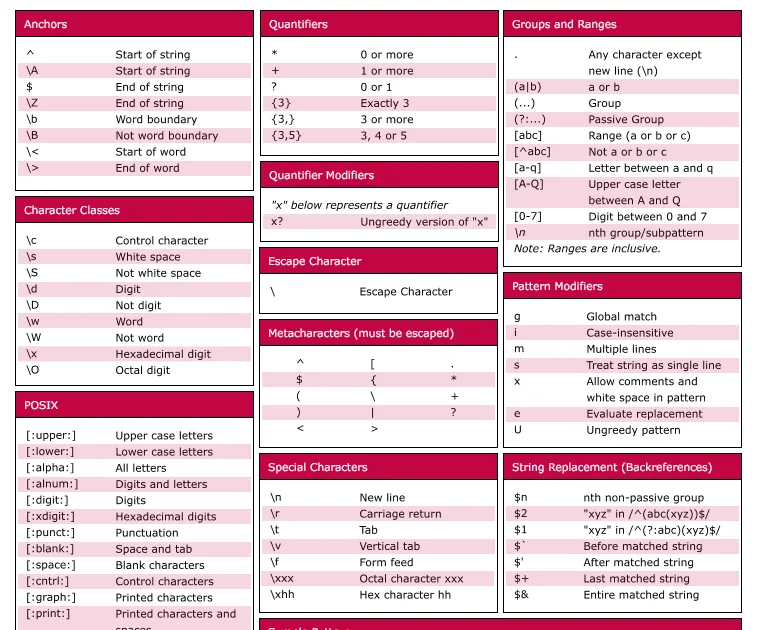
**Ключи: -v** инвертировать поиск, выдавать все строки кроме тех, что содержат шаблон

**-i** не учитывать регистр

**-n** показывать номер строки в файле

**-w** искать шаблон как слово, окружённое пробелами

**РЕГУЛЯРНЫЕ ВЫРАЖЕНИЯ:**

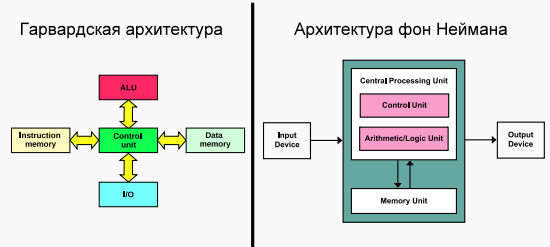


## **Задание 5**

# **Лаба 2**

### 

### **АРХИТЕКТУРЫ:**



Память - 2048 ячеек

Ячейка - 16 разрядов

### **РЕГИСТРЫ:**

1) **AR** (регистр адреса) - 11 разрядов  
2) **DR** (регистр данных) - 16 р.  
 обращение к памяти  
3) **IP** (счетчик команд) - 11 р.  
 адрес следующей исполняемой команды  
4) **ALU** (арифм-лог устройство)  
 **сложение**, **лог умножение**, **инверсия**, **прибавление единицы**  
 Коммутатор  
 (хранится в **PS**)

* **N**-Negative отрицательное число
* **Z**-Zero содержит 0
* **V**-Overflow переполнение
* **C**-Carry перенос беззнаковых чисел

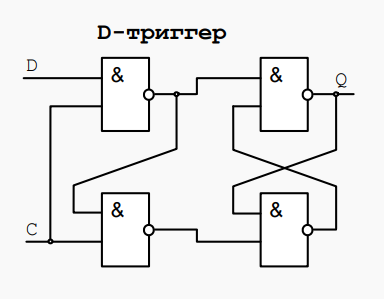
5) **BR** (буферный регистр) - 16 р.  
 промежуточное хранение данных во время работы  
6) **CR** (регистр команд) - 16 р.  
 хранение кода команды и декодирование операций  
7) **AC** (аккумулятор) – 16 р.  
 бэвм относится к эвм, где происходят вычисления с данными  
8) **SP** (указатель стека) - 11 р.  
 указ. на вершину стека, где хранятся адреса возвратов и параметров подпрограмм, прерываний  
9) **IR** (клавишный регистр) - 16 р.  
 состав пульта оператора эвм. ввод адреса программы, кодов, данных и запуска программы  
10) **PS** (регистр состояния) – 16 р.  
 хранит биты управляющие работой бэвм и признаки результата

### 

### **ТРИГГЕР:**

- устройство последовательного типа с двумя устойчивыми состояниями равновесия, предназначенных для записи и хранения информации

- ячейка информации хранящая один бит информации



### **ЦИКЛЫ КОМАНД:**

выборка команды (**IF**) instruction fetch  
выборка адреса (**AF**) address fetch  
выборка операнда (**OF**) operand fetch  
исполнение (**EX**) execution  
прерывание (**INT**) interruption

### **ЦИКЛЫ ПУЛЬТОВЫХ ОПЕРАЦИЙ:**

* **ввод адреса**
* **чтение**
* **запись**
* **пуск**

### **ДОП КОД:**

216 переводим в двоичную, делаем инверсию и прибавляем единицу  
снова переводим в 16-ричную, готово  
как проверить? Если прибавить, то будет 10000 в 16-ричной  
команда 0780  
разница: организация обращения к памяти

Гарвардская

фон Неймана

-центр устройств-Control Unit - всё в общей памяти

-память команд, данных, устройств - отдельно выделены устройства ввода-вывода

-ввода физически определены - устройство обращения к памяти едино для данных и инструкций

### **АДРЕСНЫЕ КОМАНДЫ:**

* предназначены для осуществления операция БЭВМ с использованием различных видов адресации. Вместе все 12 бит, задающие адресацию, составляют *адресную часть команды*

### 

### **БЕЗАДРЕСНЫЕ КОМАНДЫ:**

* предназначены для кодирования команд, которые не содержат адрес ячейки памяти для указания операнда или перехода. Код операции всегда равен 0000. Остальные разряды являются расширением кода операции, позволяя реализовать дополнительные безадресные команды.



### 

### **КОМАНДЫ ВЕТВЛЕНИЯ:**

* используются для организации переходов по заданному условию



### **ЛОГИЧЕСКИЕ ДЕЙСТВИЯ:**

* **дизъюнкция**
* **конъюнкция**
* **отрицание**
* **импликация**
* **эквиваленция**
* **исключающее ИЛИ**

### **ЛОГИЧЕСКИЕ ЭЛЕМЕНТЫ:**

### 

### **ДОП. КОД НА МАШИННОМ КОДЕ:** **001 0010** ищем доп код для значения 0010 **002 0200** очищаем аккумулятор **003 A001** загружаем значение 001 в аккумулятор **004 0280** инверсия аккумулятора **005 0700** инкремент +1 **006 E008** сохраняем полученное значение в ячейку 008 **007 0100** отключение **008 0001** (любое рандомное число кроме 0000)

### **КОМАНДЫ ДЛЯ НАХОЖДЕНИЯ ДОП. КОДА:**

* **Neg**
* **Not Inc**
* **Cla Sub**