Ниже текст задания Task 4 – нужно реализовать некоторые команды Unix. Чтобы выбрать номер реализуемой команды в вариативной части, поступаем следующим образом: складываем количества букв в ваших фамилии, имени, отчестве и находим остаток от деления суммы на количество задач в соответствующей группе команд, после чего добавляем единицу и получаем искомый номер. Если задача оказалась для вас слишком сложной, можно заменить на более простую, главное, чтобы сделана была самостоятельно. И наоборот, можно брать более сложные задачи для реализации, если хотите проверить свои силы.

Task 4

Нужно реализовать несколько команд Unix. Команды, обязательные для реализации:
(«реализация» означает написание соответствующей программы на языке Си; имя исполняемого файла, полученного в результате компиляции данной программы должно совпадать с именем команды; рекомендуется все команды разместить в одном подкаталоге)

1. echo – печатает свои аргументы через пробел, возможны флаги (см. man echo).
2. pwd – напечатать имя текущего каталога.
3. ls – вывести список файлов текущего каталога.

Возможен (т.е. реализовывать необязательно) список файлов-аргументов: если аргумент является каталогом, то печатается список файлов этого каталога, в противном случае печатается сам аргумент.
Возможны флаги:
-R – вывести имена всех файлов текущего каталога, а также файлов, содержащихся во вложенных подкаталогах,
-l – показать атрибуты: тип, права доступа, имя владельца, размер,
-g – показать имя группы владельца.
Если файл является символической ссылкой, желательно указывать также и имя файла, на который он ссылается, например mydir/f1 –> /user1/f2 –> /user2/subdir/f3. Здесь mydir/f1 и /user1/f2 – символические ссылки, /user2/subdir/f3 – обычный файл.

Далее перечисляются группы команд. Хотя бы одна команда из каждой группы должна быть реализована.

I. 1. mv old_file new_file (аргументы могут быть каталогами)
2. cp file copy_file
3. ln [-s] original_link new_link

II. 1. wc filename
Результат: filename кол-во_строк кол-во_слов кол-во_символов
Возможен список имен файлов; в этом случае подобная информация выдаётся о каждом файле.
2. grep substring filename
Результат: строки файла filename, содержащие substring как подстроку (возможен флаг - v; в этом случае результат – это строки, которые не содержат substring как подстроку).
3. cmp filename1 filename2
(сравнение содержимого двух файлов)
Результат: информация о первом различии.

Пример: filename1 differs from filename2: line 5 char 36
4. sort filename
(сортировка строк файла в соответствии с кодировкой ASCII)
Возможны флаги:
-r – обратный порядок,
-f – не различать большие и малые буквы,
-n – числовой порядок,
+n – начать сортировку с (n+1)-ой строки.

III. 1. cat filenames
Возможен флаг:
-n – с нумерацией строк (если файлов несколько, то нумерация сквозная)
2. tail filename
(вывод 10 последних строк файла)
Возможны флаги:
-n – n последних строк,
+n – с n-ой строки и до конца файла.
3. od filename
(вывод содержимого файла по 10 символов в строке
с указанием номера первого символа в каждой десятке)
Пример:
000001 a b c d \n e f g h i
000011 j k \t l m n
Возможен флаг:
-b – с указанием восьмеричных кодов символов.
==========

По согласованию с преподавателем можно реализовать и другие команды UNIX.

Дополнительные пояснения по Task4.

Для выполнения Task4 полезно прочитать тему 10 из [Вдовикина, Машечкин, и др. https://cmcmsu.info/download/c.programming.in.unix.pdf],
главу 8 из K&R,
материал http://www.cmcmsu.info/2course/unix.file.attributes.htm,
также полезной может быть книга [Теренс Чан. Системное программирование в Unix на C++].

По поводу реализации команд mv или cp из Task 4: чтобы понять правильно ли она работает, попробуйте взять ненужный файл f и проведите такой эксперимент:
$ mv f f, или
$ cp f f . Посмотрите, что случится с файлом f.
Команды cp и mv должны копировать не только содержимое но и (по возможности) права доступа.

Для реализации пригодятся низкоуровневые read (), write(), link(), unlink(), symlink(), stat(), fstat(), stat() и др. Смотрите справочник man.

Для реализации ls с флагами и для некоторых других команд будут полезны функции запроса атрибутов процесса (см. man):

pid_t getpid(void);

```
pid_t  getpid(void);
pid_t  getpgrp(void);
uid_t  getuid(void);
uid_t  geteuid(void);
gid_t  getgid(void);
gid_t  getegid(void).
```

Необходимо отметить, что в старых версиях UNIX все перечисленные функции возвращают значения типа int.

Изменение атрибутов процесса:

```
int  setsid(void);
int  setpgid(pid_t pid, pid_t pgid);
int  setuid(uid_t uid);
int  seteuid(uid_t uid);
int  setgid(gid_t gid);
int  setegid(gid_t gid);
```

Здесь также много полезного: https://ejudge.ru/study/3sem/unix.shtml – посмотрите номера материалов 1, 2 (там и про функцию, создающую уникальное имя для временного файла должно быть), а также номера 3 и 5.

Посмотрите в справочнике man про функции

```
char *   getcwd(char *buf, size_t size);

char *   getwd(char *buf);
```

С помощью них можно реализовать команду pwd, или с помощью getenv("PWD").

Для изменения текущей директории (рабочей директории) полезны функции:

```
int chdir(const char *path);

int  fchdir(int fildes);
```