Практикум на ЭВМ
Семестровая работа №1.
bash

Баев А.Ж.

Казахстанский филиал МГУ

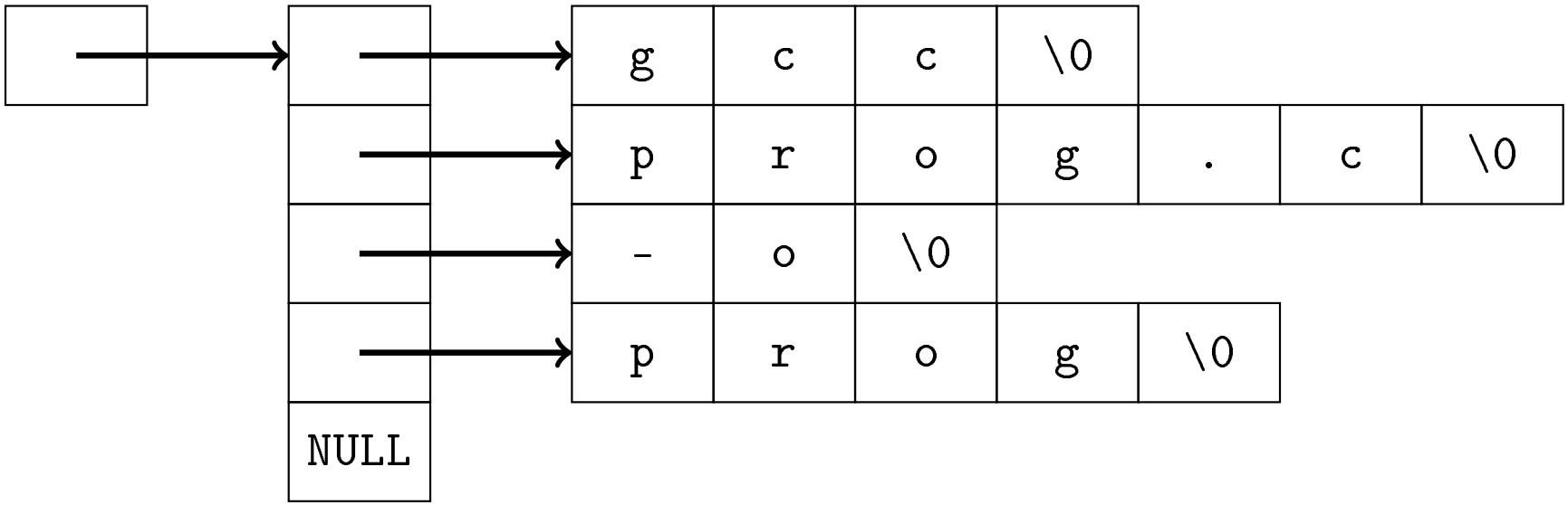
20 октября 2018

Этап 1. Делим на лексемы

Дана строка — последовательность слов, разделенных пробельными символами (табуляции, пробелы).

```
gcc prog.c -o prog
```

Создать динамический массив указателей на строки



Этап 1. Делим на лексемы

Описать функцию, которая считывает текст до пробела, табуляции или переноса строки, выделяет память и возвращает указатель на слово и на последний символ.

```
char *get_word(char *end);
```

Описать функцию, которая считывает текст до переноса строки, выделяет память и возвращает указатель

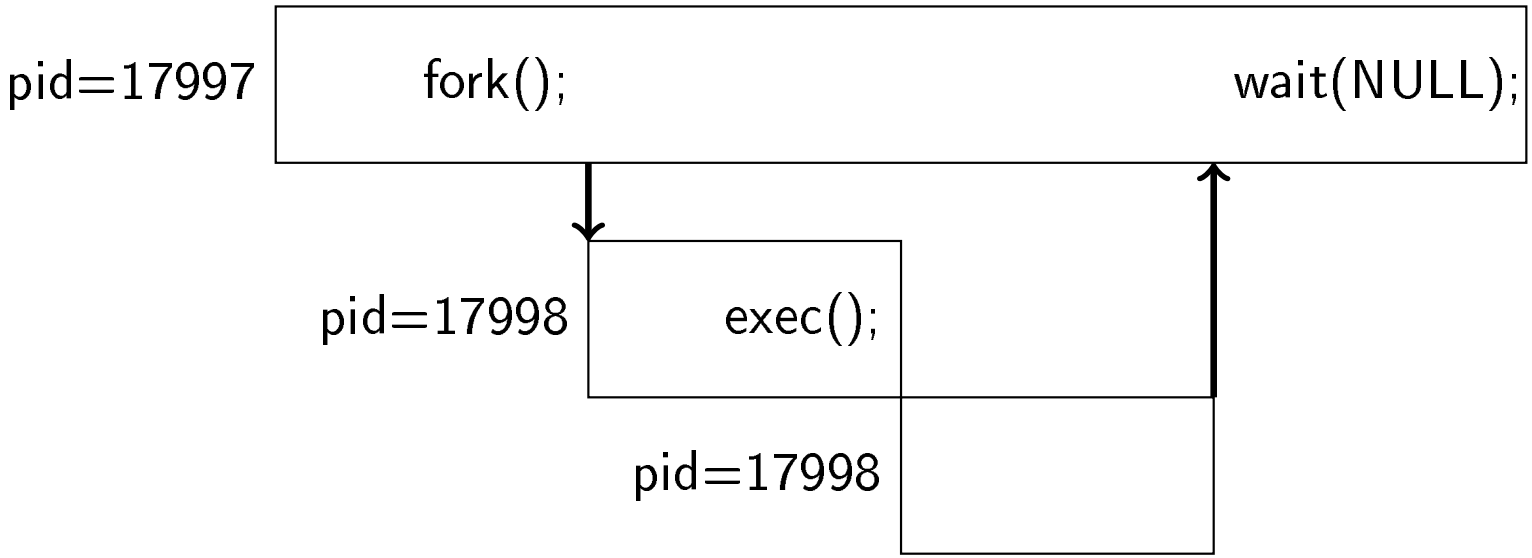
```
char **get_list();
```

Этап 2. Выполняем запуск

Необходимо в бексконечном цикле выполнять запуск программы до тех пор, пока не встретится exit или quit.

Этап 2. Выполняем запуск

```
    if (fork() > 0){ /* parent*/
        wait(NULL);
    } else { /* child */
        if (execvp(cmd[0], cmd) < 0) {
            perror("exec failed");
            return 1;
        }
}
```



Этап 3. Перенаправление ввода и вывода

Если в качестве лексемы встречается символ перенаправления ввода или вывода, то сделать соответствующий ввод или вывод в файл.

```
ls -l > f.txt
```

```
grep .c < f.txt
```

Важно: fork() копирует открытые дескрипторы и в дочке.

Этап 3. Перенаправление ввода и вывода

```
 #include <unistd.h>
int dup(int oldfd);
int dup2(int oldfd, int newfd);
```

dup() — системный вызов, который создает копию текущего файлового дескриптора (в качестве идентификатора нового дескриптора используется минимальный свободный номер).

dup2() — системный вызов, аналогичный dup(), за исключением того, что в качестве идентификатора используется указанный дескриптор newfd. Если newfd перед этим был ассоциирован с некоторым дескриптором, то он закрывается.

Этап 3. Перенаправление ввода и вывода

Дочка пишет в файл «f.txt»

```
fd = open("f.txt",
          O_WRONLY|O_CREAT|O_TRUNC,
          S_IRUSR|S_IWUSR)
if (fork() == 0) {
    dup2(fd, 1);
    execve(cmd[0], cmd)
    return 1;
}
```

Этап 4. Перенаправление вывода дочки

Перенаправим вывод дочернего процесса на стандартный ввод родительского процесса.

Этап 4. Именованный канал

```
#include <unistd.h>
int pipe(int pipefd[2]);
```

Именованный канал существует в системе и после завершения процесса.
Если в качестве лексемы встречается символ перенаправления ввода или вывода, то сделать соответствующий ввод или вывод в файл.

Этап 4. Именованный канал

```
int fd[2];
pipe(fd);
if (fork() == 0) {
    close(fd[0]);
    dup2(fd[1], 1);
    execve(cmd[0], cmd)
    return 1;
}
close(fd[1]);
dup2(fd[0], 0);
```