Практикум на ЭВМ
Семестровая работа №1.
bash

Баев А.Ж.

Казахстанский филиал МГУ

19 октября 2019

Пишем кастомный интерпретатор.

Сегодня:
1. Делим на лексемы.
2. Стандартный запуск программы.
3. Перенаправление ввода и вывода.

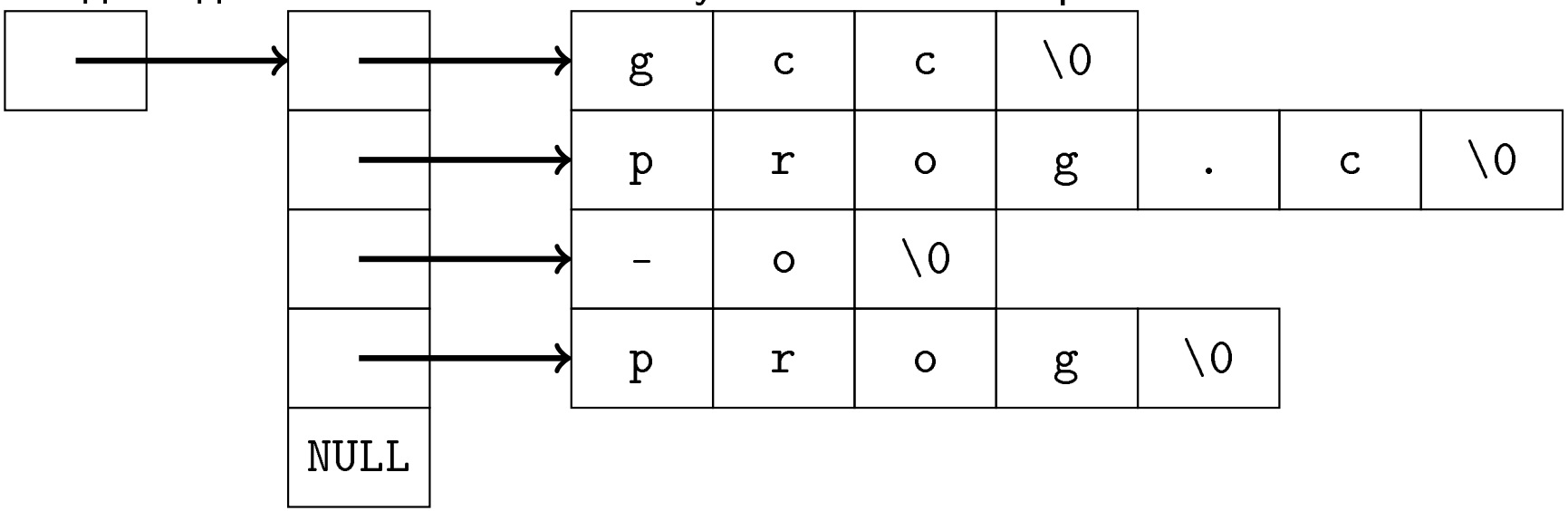
Далее: 4. Конвейер для двух элементов.
5. Конвейер для произвольного количества.
6. Фоновый режим.
7. Смена директории cd.
8. Конвейер &&.
9. Ctrl + C.

Этап 1. Делим на лексемы

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

```
1 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.

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

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

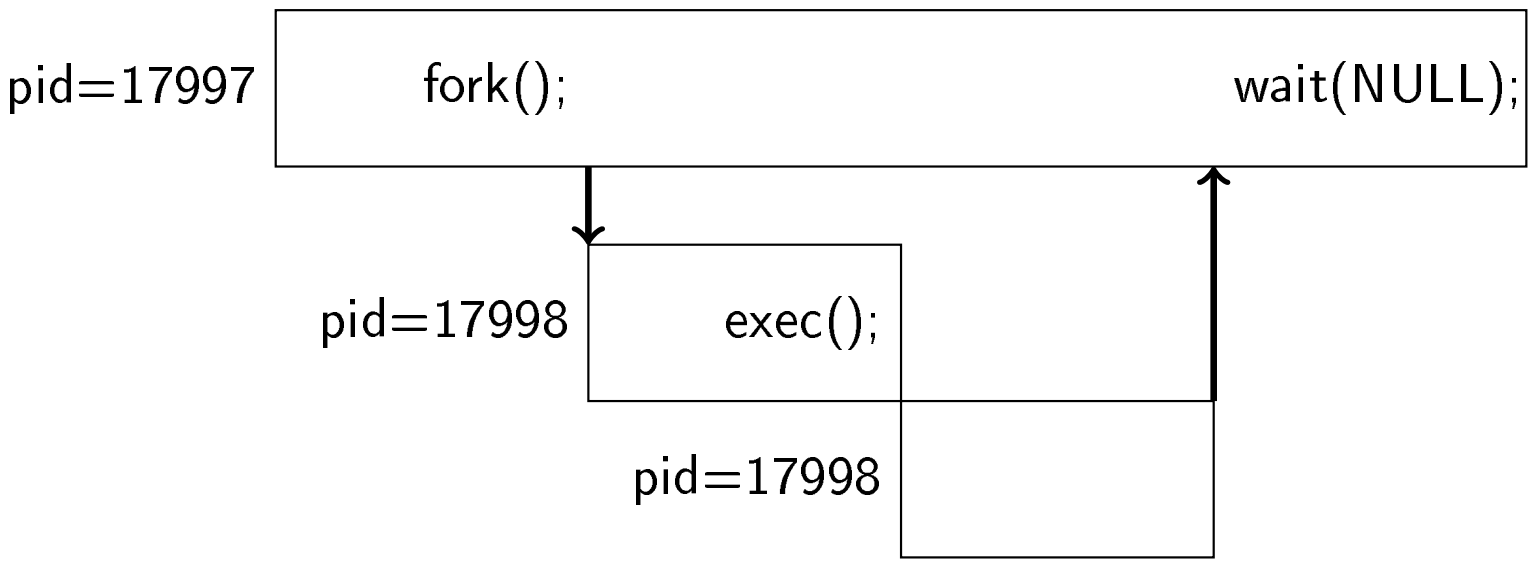
```
1 char **get_list();
```

Этап 2. Выполняем запуск

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

Этап 2. Выполняем запуск

```
1     if (fork() > 0){ /* parent*/
2         wait(NULL);
3     } else { /* child */
4         if (execvp(cmd[0], cmd) < 0) {
5             perror("exec failed");
6             return 1;
7         }
8     }
```



Этап 3. Перенаправление ввода и вывода

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

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

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

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

Этап 3. Перенаправление ввода и вывода

```
1  #include <unistd.h>
2  int dup(int oldfd);
3  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»

```
1  fd = open("f.txt",
2            O_WRONLY|O_CREAT|O_TRUNC,
3            S_IRUSR|S_IWUSR)
4  if (fork() == 0) {
5      dup2(fd, 1);
6      execve(cmd[0], cmd)
7      return 1;
8  }
```

Этап 4. Перенаправление вывода дочки

Перенаправим вывод дочернего процесса на стандартный ввод родительского процесса.

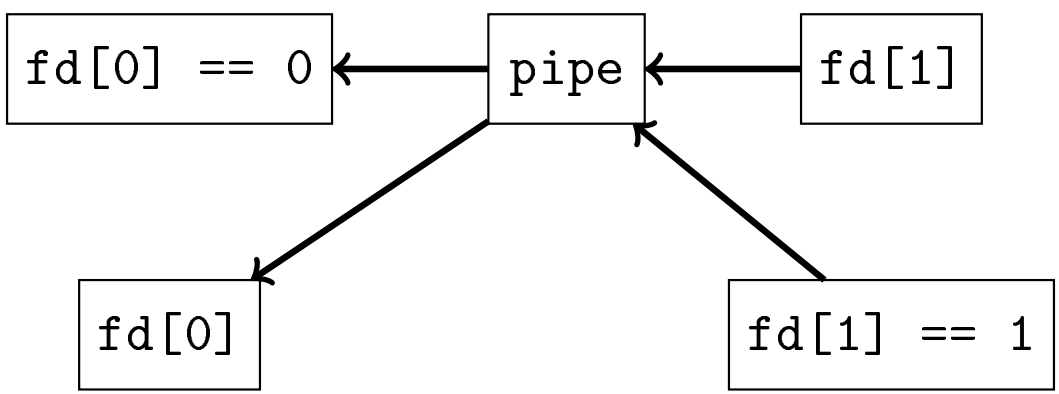
Этап 4. Неименованный канал

```
1 #include <unistd.h>
2 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. Неименованный канал

```
 1 int fd[2];
 2 pipe(fd);
 3 if (fork() == 0) {
 4     close(fd[0]);
 5     dup2(fd[1], 1);
 6     execve(cmd[0], cmd)
 7     return 1;
 8 }
 9 close(fd[1]);
10 dup2(fd[0], 0);
```



Git здорового студента

```
README.md
Makefile
sources/
  *.c
include/
  *.h


examples/
  *.c
bin/
  *.so
```

Сдача: 23/10/2019