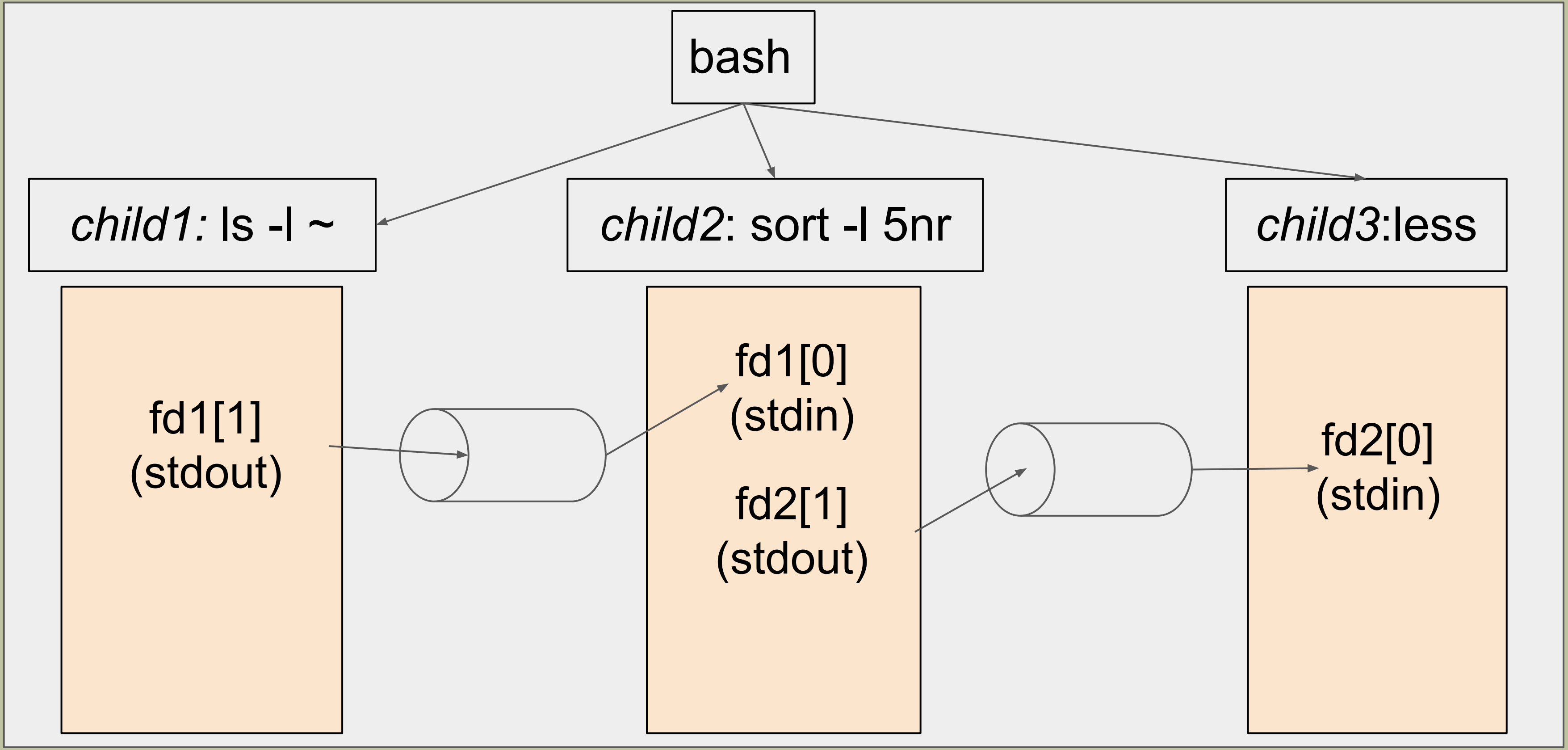
Лекция 14

- Неименованные каналы.
- Каналы FIFO (именованные каналы).
- UNIX сокеты (локальные сокеты).

```
> ls -l  ~ | sort -k 5nr | less
```

```
-rwxr-xr-x  1 malkov users 3532172082 Oct 17  2020
cuda_11.1.1_455.32.00_linux.run
-rwxr-xr-x  1 malkov users 3066694836 Jul  3  2020
cuda_11.0.2_450.51.05_linux.run
-rw-r--r--  1 malkov users   10432532 Jun 13  2021
Image-0001.png
-rw-r--r--  1 malkov users     194745 Sep 17 14:42
oscSL-W.png
-rw-r--r--  1 malkov users     183002 Sep 17 14:44
oscSL-U.png
Lines 2-6
```



bash

child1: ls -l ~

child2: sort -l 5nr

child3:less

fd1[1]
(stdout)

fd1[0]
(stdin)

fd2[1]
(stdout)

fd2[0]
(stdin)

lab14Pa-1.c

```
#include <stdio.h>
int main(){
  fprintf(stdout, "sincerely yours, ");
  return 0;
}
```

lab14Pa-2.c

```
#include <stdio.h>
#include <string.h>
#define BUFFER_SIZE  128

int main(){
  char buf[BUFFER_SIZE];
  memset(buf, 0, BUFFER_SIZE);
  fgets(buf, BUFFER_SIZE, stdin);

  buf[0]-=0x20;
  strcat(buf, "E.A. Malkov\n");

  fprintf(stdout, "%s", buf);
  return 0;
}
```

```
> ./lab14Pa-1
sincerely yours,
```

```
> ./lab14Pa-2
q
Q
E.A. Malkov
```

```
> ./lab14Pa-1  | ./lab14Pa-2
Sincerely yours, E.A. Malkov
```

```
> ./lab14Pa
Sincerely yours, E.A. Malkov
```

lab14Pa.c

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>

int main(){
  int pipe_fds[2];
  int read_fd, write_fd;

  pipe(pipe_fds);
  read_fd=pipe_fds[0];
  write_fd=pipe_fds[1];

  pid_t pid1, pid2;
```

```
pid1=fork();

if(pid1==0){
  close(write_fd);

  close(STDIN_FILENO);
  dup(read_fd);

  if(execvp("./lab14a-2",NULL)==-1)
      perror("execvp call : ");

  close(read_fd);
}
else{
  pid2=fork();
```

```
    if(pid2==0){
      close(read_fd);

      close(STDOUT_FILENO);
      dup(write_fd);

      if(execvp("./lab14a-1",NULL)==-1)
        perror("execvp call : ");

      close(write_fd);
    }
  }

  return 0;
}
```

```
> mkfifo ff_test
```

```
> ls -ltr
итого 32
-rw-r--r-- 1 malkov users   396 Dec 12 15:28 lab14Fa-c.c
-rw-r--r-- 1 malkov users   632 Dec 12 15:28 lab14Fa-s.c
-rwxr-xr-x 1 malkov users 11864 Dec 12 15:28 lab14Fa-s
-rwxr-xr-x 1 malkov users 11624 Dec 12 15:28 lab14Fa-c
prw-r--r-- 1 malkov users     0 Dec 13 18:22 ff_test
```

lab14Fa-s.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>

#define BUF_SIZE 80

int main(){
  int server_fd;
  char buf[BUF_SIZE];
  int bytes_ret;
  memset(buf,0,BUF_SIZE);
```

```
 mkfifo("Server_simple", S_IRUSR | S_IWUSR |
                    S_IWGRP);
  server_fd = open("Server_simple", O_RDONLY);

  bytes_ret=read(server_fd, buf, BUF_SIZE);
  write(fileno(stdout), buf, bytes_ret);

  //fgets(buf, BUF_SIZE, fdopen(server_fd, "r"));
  //fprintf(stdout, "%s", buf);

  unlink("Server_simple");

  close(server_fd);
  return 0;
}
```

lab14Fa-c.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>

int main(){
  int server_fd;

  mkfifo("Server_simple", S_IRUSR | S_IWUSR |
                    S_IWGRP);
  server_fd = open("Server_simple", O_WRONLY);
write(server_fd, "Hello, Server!\n",
    sizeof("Hello, Server!\n"));

  close(server_fd);

  return 0;
}
```

```
> ./lab14Fa-s
ESC
```

```
> ls -ltr
-rwxr-xr-x 1 malkov users 11864 Dec 12 15:28 lab14Fa-s
-rwxr-xr-x 1 malkov users 11624 Dec 12 15:28 lab14Fa-c
prw------- 1 malkov users     0 Dec 12 15:28 Server_simple
```

```
> ./lab14Fa-c
```

```
> ./lab14Fa-s
Hello, Server!
```

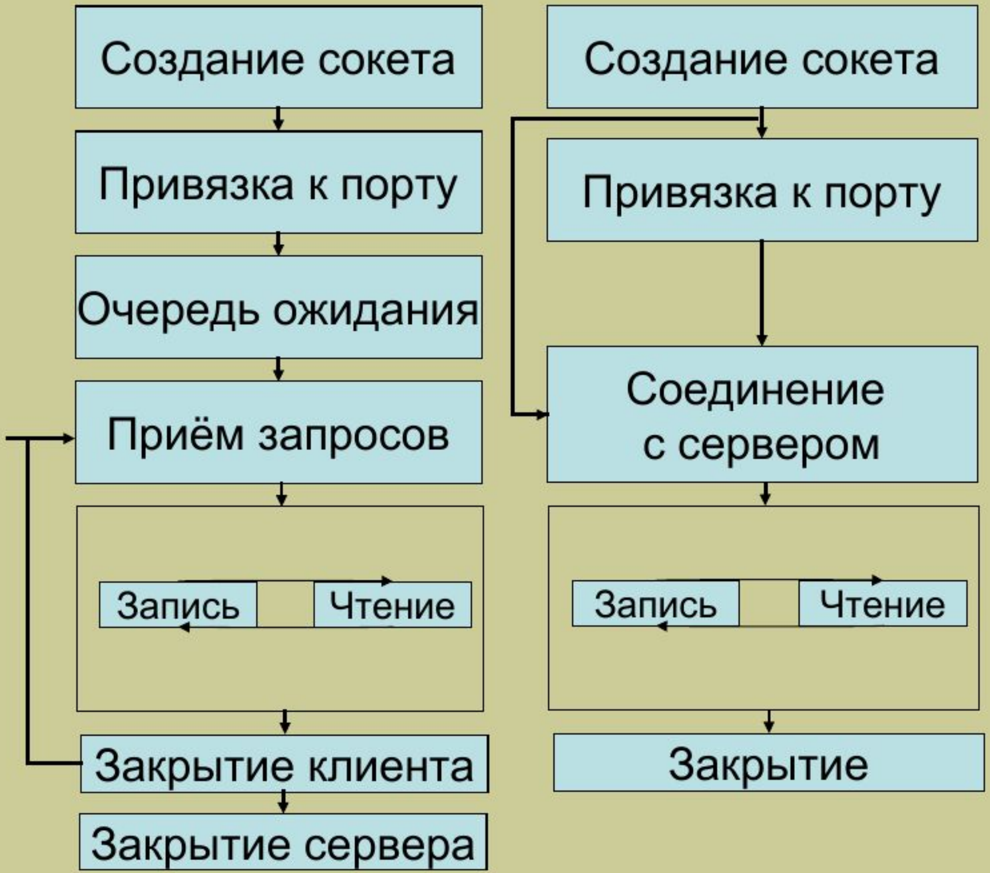
```
> ls -ltr
-rwxr-xr-x 1 malkov users 11864 Dec 12 15:28 lab14Fa-s
-rwxr-xr-x 1 malkov users 11624 Dec 12 15:28 lab14Fa-c
```

сервер

```
> cat < ff_test
Hello!
How are you?
```

клиент

```
> cat > ff_test
Hello!
How are you?
See you soon.
```



сокет–сервер

Создание сокета

Привязка к порту

Очередь ожидания

Приём запросов

Запись

Чтение

Закрытие клиента

Закрытие сервера

сокет-клиент

Создание сокета

Привязка к порту

Соединение с сервером

Запись

Чтение

Закрытие

socket()

bind()

listen()

accept()

connect()

write()
read()

close()

lab14S-s.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <unistd.h>

int srv(int cln_sock);

int main(int argc, char** argv){
  if(argc<2){
    fprintf(stderr, "USAGE: prog <socket name> ");
    return -1;
  }
```

```
const char* const sock_name=argv[1];
int sock_fd;
struct sockaddr_un  name;
int cln_sent_quit_message;

sock_fd=socket(PF_LOCAL, SOCK_STREAM, 0);
name.sun_family=AF_LOCAL;
strcpy(name.sun_path, sock_name);
bind(sock_fd, (struct sockaddr *)&name, SUN_LEN(&name));

listen(sock_fd, 5);
```

```
do{
  struct sockaddr_un  cln_name;
  socklen_t cln_name_len;
  int cln_sock_fd;

  cln_sock_fd=accept(sock_fd,
                     (struct sockaddr *)&cln_name,
                     &cln_name_len);
  cln_sent_quit_message=srv(cln_sock_fd);
  close(cln_sock_fd);
}
while(!cln_sent_quit_message);
```

```
  close(sock_fd);
  unlink(sock_name);

  return 0;
}
```

```
int srv(int cln_sock){
  while(1){
    int length;
    char* text;
    if(read(cln_sock, &length, sizeof(length))==0)
      return 0;
    text=(char*)malloc(length);
    read(cln_sock, text, length);
    fprintf(stdout,"%s\n", text);

    if(!strcmp(text, "quit")){
      free(text);
      return 1;
    }

    free(text);
  }
}
```

lab14S-c.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <unistd.h>

void write_text(int , const char* );

int main(int argc, char** argv){
  if(argc<3){
    fprintf(stderr, "USAGE: prog <socket name> <message>");
    return -1;
  }
  const char* const sock_name=argv[1];
  const char* const message=argv[2];
  int sock_fd;
  struct sockaddr_un  name;

  sock_fd=socket(PF_LOCAL, SOCK_STREAM, 0);
  name.sun_family=AF_LOCAL;
  strcpy(name.sun_path, sock_name);

  connect(sock_fd, (struct sockaddr *)&name, SUN_LEN(&name));
  write_text(sock_fd, message);

  close(sock_fd);
  return 0;
}
```

```
void write_text(int sock_fd, const char* text){
  int txt_length=strlen(text)+1;

  write(sock_fd, &txt_length, sizeof(txt_length));
  write(sock_fd, text, txt_length);
}
```