Лекция 4

- Системный вызов fork() (продолжение).
- Прекращение выполнения процесса. Zombie.
- Системные вызовы exec*.

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

int main(){
 pid_t child_pid;

 child_pid=fork();
```

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

int main(){
 pid_t child_pid;
 int  child_status;

 child_pid=fork();
 if( child_pid!=0)
   pause();

 return 0;
}
```

```
 if( child_pid!=0){
   wait(&child_status);
   pause();
 }

 return 0;
}
```

```
~> ps -e -o pid,ppid,pgid,sid,state,command  | grep 3294
```

```
 3294  3224  3294  3294 S /bin/bash
 4743  3294  4743  3294 S ./pz
 4744  4743  4743  3294 Z [pz] <defunct>
```

```
~> kill 4744
~> ps -e -o pid,command | grep pz
```

```
4743 ./pz
 4744 [pz] <defunct>
```

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

int main(){
 pid_t child_pid;
 int child_status;

 child_pid=fork();
```

```
if( child_pid==0){
   fprintf(stdout, "%d\n", getpid());
   fprintf(stdout, "%d\n", getppid());
   fprintf(stdout, "%d\n", getpgid(getpid()));
   fprintf(stdout, "%d\n", getsid(getpid()));
 }
 else if( child_pid!=0){
   wait(&child_status);
   fprintf(stdout, "\n\n%d\n", getpid());
   fprintf(stdout, "%d\n", getppid());
   fprintf(stdout, "%d\n", getpgid(getpid()));
   fprintf(stdout, "%d\n", getsid(getpid()));
 }
 return 0;
}
```

```
~>./pw
8599    8598    8598    3294
8598    3294    8598    3294
```

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

void oldman(){
  fprintf(stdout, "I'm not yet dead! ID is %i\n", (int) getpid());
}
void recreation(){
  fprintf(stdout, "Who I am? ID is %i\n", (int) getpid());
}
```

```
int main(){
  pid_t child_pid, parent_pid;
  int i=0;

  parent_pid=(int) getpid();

  child_pid=fork();
```

```
while(i++<5)
 if(child_pid!=0){
   oldman();
   usleep(100);
   if(i==3) kill(child_pid,SIGTERM);
 }
 else{
  recreation();
  usleep(100);
}
```

```
  if(child_pid!=0) pause();
  return 0;
}
```

I'm not yet dead! My ID is 11625
Who I am? My ID is 11626
I'm not yet dead! My ID is 11625
I'm not yet dead! My ID is 11625
Who I am? My ID is 11626
I'm not yet dead! My ID is 11625
I'm not yet dead! My ID is 11625

I'm not yet dead! My ID is 11639
Who I am? My ID is 11640
I'm not yet dead! My ID is 11639
Who I am? My ID is 11640
I'm not yet dead! My ID is 11639
Who I am? My ID is 11640
I'm not yet dead! My ID is 11639
I'm not yet dead! My ID is 11639

```
~> ./pk
I'm not yet dead! ID is 4429
Who I am? ID is 4430
I'm not yet dead! ID is 4429
I'm not yet dead! ID is 4429
Who I am? ID is 4430
I'm not yet dead! ID is 4429
I'm not yet dead! ID is 4429
|
```

```
~> ps -e -o pid,ppid,pgid,sid,state,command | grep pk
 4429  3241  4429  3241 S ./pk
 4430  4429  4429  3241 Z [pk] <defunct>
```

Создание процессов с помощью семейства системных вызовов exec*.

l4.c

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

int main(int argc, char* argv[]){
  if(execvp(argv[1], argv)==-1)
    perror("execvp call : ");

  fprintf(stdout, "Everything is ignored!\n");
  return 0;
}
```

sinf.c

```
#include <stdio.h>
#include <sys/sysinfo.h>

int main(){
 const long minute = 60;
 const long hour = minute*60;
 const long day = hour*24;

 struct sysinfo si;
 sysinfo(&si);

 printf("system uptime : %ld days, %ld:%02ld:%02ld\n",
                  si.uptime/day, (si.uptime % day)/ hour,
                  (si.uptime % hour)/minute, si.uptime % minute);

 printf("total RAM  : %d KB\n", si.totalram/1024);
 printf("free RAM   : %d KB\n", si.freeram /1024 );
 printf("total SWAP : %d KB\n", si.totalswap / 1024);
 printf("free  SWAP : %d KB\n", si.freeswap / 1024);
 printf("process count : %d\n", si.procs);

 return 0;
}
```

```
~Lab4> ./l4 sinf
execvp call : : No such file or directory
Everything is ignored!
```

```
~Lab4> ./l4 ../Lab2/sinf
system uptime : 0 days, 0:18:27
total RAM   : 16313772 KB
free RAM    : 10449628 KB
total SWAP : 16777212 KB
free  SWAP : 16777212 KB
process count : 1234
```

```
…/Lab4> echo $PATH
/usr/local/cuda-11.2/bin:/home/malkov/anaconda3/bin:/home/malkov/
anaconda3/condabin:/usr/local/cuda-11.2/bin:/home/malkov/bin:/usr/l
ocal/bin:/usr/bin:/bin:/snap/bin
```

```
…/Lab4> export PATH=$PATH:../Lab2
```

```
/Lab4> echo $PATH
…/usr/bin:/bin:/snap/bin:../Lab2
```

```
…/Lab4> ./l4 sinf
system uptime : 0 days, 0:38:48
total RAM   : 16313772 KB
free RAM    : 10462360 KB
total SWAP : 16777212 KB
……………………………………
```

```
…/Lab2> cat ~/.bashrc
export
PATH=/usr/local/cuda-11.2/bin${PATH:+:${PATH}}
export
LD_LIBRARY_PATH=/usr/local/cuda-11.2/lib64${LD_LIBRARY_
PATH:+:${LD_LIBRARY_PATH}}

export PATH="/home/malkov/anaconda3/bin:$PATH"
……………………………………………………………
```

```
int execl(const char *path, const char *arg, ...);
int execlp(const char *file, const char *arg, ...);
int execle(const char *path, const char *arg,..., char * const envp[]);
int execv(const char *path, char *const argv[]);
int execvp(const char *file, char *const argv[]);
int execvpe(const char *file, char *const argv[],char *const envp[]);
```