ITMO University

The Faculty of Software Engineering and Computer Systems

System software basics

Practical task №5

Done by
a student of P3310 group
Dima Glushkov

Saint-Petersburg
2019

Part 1.

predefines.h

```
#ifndef PREDEFINES_H
#define PREDEFINES_H

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

const int DELAY = 1;
const key_t KEY = 200;
const char * SHAREDFILE = "/tmp/shared";

typedef struct {
    pid_t pid;
    gid_t gid;
    uid_t uid;
    double loadavg[3];
    time_t time;
} sys_info;

typedef struct {
    long type;
    sys_info info;
} message;

void print_info(sys_info * info)
{
    puts("Server info:");
    printf("pid: %d; uid: %d; gid: %d\n", info->pid, info->uid, info->gid);
    printf("uptime: %ld\n", info->time);
    printf("loadavg in 1 min: %.3f; 5 min: %.3f; 15 min: %.3f\n", info-
>loadavg[0], info->loadavg[1], info->loadavg[2]);
}

sys_info init_info()
{
    sys_info info;
    info.pid = getpid();
    info.uid = getuid();
    info.gid = getgid();
    info.time = 0;
    getloadavg(info.loadavg, 3);

    return info;
}

#endif
```

Task 1.
Client

```
#include <stdio.h>
#include <sys/shm.h>
#include <sys/ipc.h>
#include <unistd.h>
#include <string.h>
#include "predefines.h"

int main()
{
      sys_info * info;
      int mem_id;
      mem_id = shmget(KEY, sizeof(info), 0);
      info = (sys_info*) shmat(mem_id, NULL, 0);
      print_info(info);
      return 0;
}
```

Server

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <assert.h>
#include <time.h>
#include <string.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include "predefines.h"

sys_info init_info();

int main(void)
{
    int mem_id;
    sys_info * info;
      time_t start_time = time(NULL);
      time_t cur_time;



      if((mem_id = shmget(KEY, sizeof(info), IPC_CREAT | 0644)) < 0)
      {
            perror("Error while shmget()");
            exit(1);
      }

      if((info = (sys_info*)shmat(mem_id, NULL, 0)) == NULL)
      {
        perror("Error while shmat()");
            exit(1);
      }

      *info = init_info();

      while(1)
      {
```

```
            cur_time = time(NULL);
            assert(cur_time > 0);

            info->time = cur_time - start_time;
            assert(getloadavg(info->loadavg, 3));
            sleep(DELAY);
      }

}
```

Task 3.

Client

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <fcntl.h>
#include <sys/mman.h>
#include <unistd.h>
#include <string.h>
#include "predefines.h"

int main()
{
      int fd;
      sys_info * info;

      if((fd = open(SHAREDFILE, O_RDWR)) < 0)
      {
            perror("Error while open90");
            exit(1);
      }


      if((info = (sys_info*)mmap(NULL, sizeof(sys_info), PROT_READ, MAP_SHARED,
fd, 0)) == MAP_FAILED) {
            perror("Error while mmap()");
            exit (1);
      }

    print_info(info);

      munmap(info, sizeof(sys_info));
      return 0;
}
```

Server

```
#include <stdio.h>
#include <unistd.h>
#include <assert.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/mman.h>
#include <fcntl.h>
#include <time.h>
#include <string.h>
#include <stdlib.h>
#include "predefines.h"

int main()
```

```
{
    int fd;
      sys_info * serv_info;
      time_t start_time = time(NULL);
      time_t cur_time;

      unlink(SHAREDFILE);

      if((fd = open(SHAREDFILE, O_RDWR | O_CREAT, S_IREAD | S_IWRITE)) < 0) {
            perror("Error while open()");
            exit (1);
      }

      if((serv_info = (sys_info*)mmap(NULL, sizeof(sys_info), PROT_READ|
PROT_WRITE, MAP_SHARED, fd, 0)) == MAP_FAILED) {
            perror("Error while mmap()");
            exit (1);
      }


      truncate(SHAREDFILE, sizeof(sys_info));

      *serv_info = init_info();

      while(1)
      {
            cur_time = time(NULL);
            serv_info->time = cur_time - start_time;
            assert(getloadavg(serv_info->loadavg, 3));
            sleep(1);
      }
}
```