Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Институт «Электронных и информационных систем»

Кафедра «Информационных систем и технологий»

**СЕМЕЙСТВО ПРОТОКОЛОВ TCP/IP. СОКЕТЫ В UNIX И РАБОТА С НИМИ**

Лабораторная работа №11 по учебной дисциплине «Операционные системы»

По направлению 09.03.01 «Информатика и вычислительная техника»

Отчёт

Принял преподаватель:

\_\_\_\_\_\_\_\_\_ Ананьев В. В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

Выполнил студент группы 8091:

\_\_\_\_\_\_\_ Бочкарев Б. А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

Великий Новгород

2020

**Цель работы:** научиться работать с сокетами в UNIX-like системах.

**Задание:**

Написать программу-клиент и программу-сервер для выполнения следующего сценария: клиент генерирует случайным образом массив целых чисел с заданным размером и диапазоном значений; клиент передаёт по сети массив на сервер; сервер сортирует массив по убыванию и возвращает его клиенту; клиент выводит на экран отсортированный массив и время ожидания ответа от сервера.

Вариант задания определяется по номеру студента по списку группы на сайте [www.novsu.ru](http://www.novsu.ru) .

Протокол транспортного уровня: чётные номера – UDP, нечётные – TCP.

При запуске программы-сервера через параметры командной строки передаётся порт, на котором будет работать сервер. Если данный порт занят, программа-сервер должна получить порт по усмотрению системы. При успешном запуске сервер должен выдать на экран номер порта, на котором он работает.

При запуске программы-клиента через параметры командной строки передаётся IP-адрес и порт (сокет) программы-сервера в формате <IP-address>:<port> (например, 192.168.0.100:4242). После запуска программы-клиента пользователь вводит размер массива, минимальное и максимальное значение элементов.

**Реализация:**

**Server.c:**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <time.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <errno.h>

#include <locale.h>

int compare\_func(const void \*a, const void \*b)

{

return \*((int \*)b) - \*((int \*)a);

}

int main(int argc, char\* argv[])

{

setlocale(LC\_ALL, "Russian");

int port = atoi(argv[1]);

char message[1024];

struct sockaddr\_in server\_addr, client\_addr;

memset(&server\_addr, 0, sizeof(server\_addr));

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

server\_addr.sin\_port = htons(port);

int sock = socket(PF\_INET, SOCK\_DGRAM, 0);

if (sock < 0)

{

perror("Ошибка сокета");

return -1;

}

if (bind(sock, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) < 0)

{

server\_addr.sin\_port = 0;

if (bind(sock, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) < 0)

{

perror("Ошибка бинда");

return -1;

}

}

socklen\_t server\_len = sizeof(server\_addr);

getsockname(sock, (struct sockaddr \*)&server\_addr, &server\_len);

printf("Прослушивание порта: %d\n", ntohs(server\_addr.sin\_port));

while (1)

{

socklen\_t client\_len = sizeof(client\_addr);

int n = recvfrom(sock, message, 1024, 0, (struct sockaddr \*)&client\_addr, &client\_len);

qsort(message, n / sizeof(int), sizeof(int), compare\_func);

sendto(sock, message, n \* sizeof(int), 0, (struct sockaddr \*)&client\_addr, client\_len);

}

}

**Client.c:**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <time.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <errno.h>

#include <unistd.h>

#include <locale.h>

void take\_ip(char\* args, char\* ip, size\_t pointer)

{

memset(ip, 0, pointer + 1);

for (size\_t i = 0; i < pointer; ++i)

{

ip[i] = args[i];

}

}

void random\_digits(int\* digits, int size, int mind, int maxd)

{

srand(time(NULL));

for (size\_t i = 0; i < size; ++i)

{

digits[i] = mind + rand() % (maxd - mind + 1);

}

}

int main(int argc, char\* argv[])

{

setlocale(LC\_ALL, "Russian");

char\* args = argv[1];

char\* port = strchr(args, ':');

size\_t pointer = port - args;

char ip[pointer + 1];

take\_ip(args, ip, pointer);

printf("%s:%s\n", ip, port + 1);

int size, mind, maxd;

int digits[size];

printf("Укажите размер и диапазон (например, 10 1 10): ");

scanf("%d %d %d", &size, &mind, &maxd);

random\_digits(digits, size, mind, maxd);

printf("Вами сгенерированы следующие числа: ");

for (size\_t i = 0; i < size; ++i)

{

printf("%d ", digits[i]);

}

printf("\n");

struct sockaddr\_in server\_addr, client\_addr;

int sock = socket(PF\_INET, SOCK\_DGRAM, 0);

if (sock < 0)

{

perror("Ошибка сокета");

return -1;

}

memset(&client\_addr, 0, sizeof(client\_addr));

memset(&server\_addr, 0, sizeof(server\_addr));

client\_addr.sin\_family = AF\_INET;

client\_addr.sin\_port = htons(0);

client\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

if (bind(sock, (struct sockaddr \*)&client\_addr, sizeof(client\_addr)) < 0)

{

perror("Ошибка бинда");

return -1;

}

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(atoi(port + 1));

inet\_aton(ip, &server\_addr.sin\_addr);

time\_t startTime = time(NULL);

int n = sendto(sock, digits, size \* sizeof(int), 0, (struct sockaddr \*)&server\_addr, sizeof(server\_addr));

recvfrom(sock, digits, size \* sizeof(int), 0, (struct sockaddr \*)NULL, NULL);

time\_t endTime = time(NULL);

printf("Обработанный массив чисел:\n");

for (size\_t i = 0; i < size; ++i)

{

printf("%d ", digits[i]);

}

printf("\n");

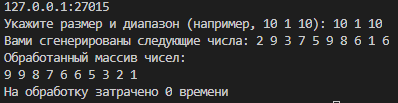
printf("На обработку затрачено %ld времени\n", endTime - startTime);

close(sock);

return 0;

}

**Результат выполнения программы:**



На обработку затрачено 0 секунд, поскольку процесс обработки занимает <1 секунды.

**Вывод:** В процессе выполнения лабораторной работы я научился работать с сокетами в UNIX-like системах.