Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования "Новгородский государственный университет имени Ярослава Мудрого"

Кафедра «Информационных технологий и систем»

Дисциплина «Операционные системы»

Отчет по лабораторной работе

«Семейство протоколов TCP/IP. Сокеты в UNIX и работа с ними »

Выполнил студент группы 9091

\_\_\_\_\_\_\_\_\_\_\_\_\_/Юшин Григорий Станиславович/

Подпись ФИО

Принял преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_/Ананьев Владислав Валерьевич/

Подпись ФИО

Великий Новгород

2021

**Цель лабораторной работы**

## Задание на лабораторную работу

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.

Протокол транспортного уровня: чётные номера**– 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.

**Исходный текст программ**

|  |
| --- |
| server.c |
| #include <sys/types.h>  #include <sys/socket.h>  #include <netinet/in.h>  #include <arpa/inet.h>  #include <string.h>  #include <stdio.h>  #include <errno.h>  #include <unistd.h>  #include <stdlib.h>  void sortArraySM(int \*mem, int ArraySize)  {  int temp;  for (int i = 0; i < ArraySize - 1; i++)  {  for (int j = 0; j < ArraySize - i - 1; j++)  {  if (mem[j] < mem[j + 1])  {  temp = mem[j];  mem[j] = mem[j + 1];  mem[j + 1] = temp;  };  };  };  }  void main(int argc, char \*argv[])  {  int ArraySize;  int listenSock\_desc, clientSock\_desc;  int clien\_len;  int recieved\_n;  struct sockaddr\_in servaddr, cliaddr;  if((listenSock\_desc = socket(AF\_INET, SOCK\_STREAM, 0)) < 0){  perror(NULL);  exit(1);  }  bzero(&servaddr, sizeof(servaddr));  servaddr.sin\_family= AF\_INET; servaddr.sin\_port= htons(atoi(argv[1])); servaddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);  if(bind(listenSock\_desc, (struct sockaddr \*) &servaddr,sizeof(servaddr)) < 0)  {  servaddr.sin\_port = 0;  if(bind(listenSock\_desc, (struct sockaddr \*) &servaddr,sizeof(servaddr)) < 0)  {  perror(NULL);  close(listenSock\_desc);  exit(1);  }  }  if(listen(listenSock\_desc, 5) < 0){  perror(NULL);  close(listenSock\_desc);  exit(1);  }  socklen\_t lengar = sizeof(servaddr);  getsockname(listenSock\_desc, (struct sockaddr \*)&servaddr, &lengar);  printf("Server port: %d\n\n", ntohs(servaddr.sin\_port));  while(1)  {  clien\_len = sizeof(cliaddr);  if((clientSock\_desc = accept(listenSock\_desc,(struct sockaddr \*) &cliaddr, &clien\_len)) < 0)  {  perror(NULL);  close(listenSock\_desc);  exit(1);  }  read(clientSock\_desc, &ArraySize, 4);  ArraySize = ntohl(ArraySize);  int opArray[ArraySize];  int temp;  for(int i = 0; i<ArraySize; i++)  {  read(clientSock\_desc, &temp, sizeof(int));  temp = ntohl(temp);  opArray[i]=temp;  }  sortArraySM(opArray,ArraySize);  for(int i = 0; i < ArraySize; i++)  {  temp = htonl(opArray[i]);  write(clientSock\_desc, &temp,sizeof(int));  }  close(clientSock\_desc);  }  close(listenSock\_desc);  } |
| Client.c |
| #include <sys/types.h>  #include <sys/socket.h>  #include <netinet/in.h>  #include <arpa/inet.h>  #include <string.h>  #include <stdio.h>  #include <errno.h>  #include <unistd.h>  #include <stdlib.h>  #include <string.h>  void fillArrayRangedSem(int \*ShMemPtr, int ArraySize, int Max, int Min)  {  for(int i = 0; i < ArraySize; i++)  {  \*ShMemPtr = (rand() % (Max - Min + 1)) + Min;  printf("%i is %i\n", i+1, \*ShMemPtr);  ShMemPtr++;  };  printf("Filled array with %i pseudo-rand int\n", ArraySize);  }  void printArray(int \*ShMemPointer, int ArraySize)  {  printf("Sorted Array\n");  for(int i = 0; i < ArraySize; i++)  {  printf("%i is %i\n", i+1, \*ShMemPointer);  ShMemPointer++;  };  }  void main(int argc, char \*\*argv)  {  int ArraySize, ArrayMax, ArrayMin;  scanf("%d%d%d", &ArraySize, &ArrayMin, &ArrayMax);  int opArray[ArraySize];  fillArrayRangedSem(opArray, ArraySize, ArrayMax, ArrayMin);  int socket\_desc;  struct sockaddr\_in servaddr;  if(argc != 2){  printf("Usage: a.out <IP address>:<port>\n");  exit(1);  }  char temp\_port[5];  int temp\_ctr=1;  for(; argv[1][temp\_ctr-1] != ':';)  {  temp\_ctr++;  }  char ip\_adr[temp\_ctr];  memset(ip\_adr,0,sizeof ip\_adr);  strncpy(ip\_adr,argv[1],temp\_ctr-1);  printf("%s\n",ip\_adr);  for(int i = 0; temp\_ctr < strlen(argv[1]); i++)  {  temp\_port[i] = argv[1][temp\_ctr];  temp\_ctr++;  }  printf("%s\n",temp\_port);  int port;  port = atoi(temp\_port);  if((socket\_desc = socket(PF\_INET, SOCK\_STREAM, 0)) < 0)  {  perror(NULL);  exit(1);  }  bzero(&servaddr, sizeof(servaddr));  servaddr.sin\_family = AF\_INET; servaddr.sin\_port = htons(port);  if(inet\_aton(ip\_adr, &servaddr.sin\_addr) == 0)  {  printf("Invalid IP address\n");  close(socket\_desc);  exit(1);  }  if(connect(socket\_desc, (struct sockaddr \*) &servaddr,  sizeof(servaddr)) < 0){  perror(NULL);  close(socket\_desc);  exit(1);  }  int intBuf = htonl(ArraySize);  write(socket\_desc,&intBuf,sizeof(int));  for(int i = 0; i < ArraySize; i++)  {  intBuf = htonl(opArray[i]);  write(socket\_desc,&intBuf,sizeof(int));  };  for(int i = 0; i < ArraySize; i++)  {  read(socket\_desc,&intBuf,sizeof(int));  opArray[i] = ntohl(intBuf);  }  printArray(opArray,ArraySize);  close(socket\_desc);  } |

**Результат выполнения программ**

|  |
| --- |
| /server 2406 |
| Server port: 2406 |

|  |
| --- |
| /server 23 |
| Server port: 41217 |

|  |
| --- |
| /client 127.0.1.0:41217 |
| [xhaver@localhost Lab11]$ ./client 127.0.0.1:41217  10 1 15  1 is 14  2 is 2  3 is 13  4 is 11  5 is 9  6 is 11  7 is 2  8 is 13  9 is 10  10 is 2  Filled array with 10 pseudo-rand int  Sorted Array  1 is 14  2 is 13  3 is 13  4 is 11  5 is 11  6 is 10  7 is 9  8 is 2  9 is 2  10 is 2 |