Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Кафедра информационных технологий и систем

**Сетевая игра «Одиннадцать предметов»**

Курсовая работа

по дисциплине:

«Операционные системы»

Разработал:

Студент группы 9091

Семёнов Егор Сергеевич

«\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2020г.

Проверил:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2020г.

**Великий Новгород**

**2020**

**ОГЛАВЛЕНИЕ**

ВВЕДЕНИЕ.................................................................................................................................... 3

ГЛАВА 1. РАЗРАБОТКА ПРОЕКТА............................................................................................ 4

1.1 Постановка задачи................................................................................................................... 4

1.2 Выбор инструментальных средств........................................................................................ 4

1.3 Выбор модели.......................................................................................................................... 4

1.4 Алгоритм решения задачи.......................................................................................................4

ГЛАВА 2. РЕАЛИЗАЦИЯ ПРОЕКТА......................................................................................... 5

2.1 Структура клиента и сервера................................................................................................. 5

2.2 Программный интерфейс сервера......................................................................................... 5

2.3 Описание программной реализации клиента....................................................................... 5

2.4 Системные вызовы.................................................................................................................. 5

2.5 Пример работы приложения................................................................................................... 6

2.6 Результат работы...................................................................................................................... 6

ЗАКЛЮЧЕНИЕ.............................................................................................................................. 7

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ....................................................................... 8

Листинг В1 — Код из файла server.с…........................................................................................ 9

Листинг В2 — Код из файла client.с…......................................................................................... 14

**ВВЕДЕНИЕ**

Цель данной работы – создание простейшей консольной версии игры «Одиннадцать предметов».

**Одиннадцать предметов**— игра для двух людей, связанная со смекалкой игроков и предварительного расчета. Игра будет работать под операционной системой Linux.

Правила игры:

В игре участвуют только два игрока.

На столе — одиннадцать предметов, например, спичек. Первый играющий берет себе из этого количества по своему усмотрению 1, 2 или 3 спички. Затем второй играющий берет себе из числа оставшихся спичек также по своему усмотрению 1, 2 или 3. Потом опять берет первый и так далее. Так поочередно оба играющих берут каждый раз не более чем по три спички. Выигрывает тот, кто берет последнюю (или последние) спичку.

Функции, реализованные в игре:

1. Подключение двух игроков
2. Возможность совершать ходы
3. Консольный интерфейс для ввода хода и вывода результата

**ГЛАВА 1. РАЗРАБОТКА ПРОЕКТА**

**1.1 Постановка задачи**

Задачей данной курсовой работы является разработка сетевой игры «Одиннадцать предметов», которая рассчитана на двух игроков. Игра должна работать под управлением операционной системы семейства Unix/Linux. Сетевая часть игры должна быть выполнена на языке C++ с помощью сокетов предоставляемых системной библиотекой <sys/socket.h>.

**1.2 Выбор инструментальных средств**

Для реализации сетевого взаимодействия используются протокол передачи данных: TCP.

**1.3 Выбор модели**

Для взаимодействия между игроками выбрана клиент-серверная модель.

**1.4 Алгоритм решения задачи**

Подключение игроков происходит следующим образом:

К серверу подключаются первый и второй игрок, введя адрес сервера и номер порта.

Ход игры:

Сервер ожидает ходы от обоих игроков. Сервер отправляет коды игрокам о том, кто ходит, после чего ждет от них ответа. При получении ответа сервер убирает указанное количество предметов со стола и проверяет остались ли они ещё, если нет объявляется победитель, иначе игра продолжается.

**ГЛАВА 2. РЕАЛИЗАЦИЯ ПРОЕКТА**

**2.1 Структура клиента и сервера**

Соединение между клиентом и сервером. Для игры также используются некоторые дополнительные функции.

**2.2 Программный интерфейс сервера**

Для обмена информацией между клиентом и сервером используются специальные коды: 1 – взять один предмет, 2 – взять два предмета, 3 – взять три предмета. Клиент же серверу отправляет следующие коды: 0 – ход игрока, 1 – победа игрока, 2 – победа противника.

**2.3 Описание программной реализации клиента**

При запуске клиента происходит подключение к серверу. Коды о его состоянии он получает от сервера, в зависимости от которых выводятся соответствующие сообщения.

**2.4 Системные вызовы**

|  |  |
| --- | --- |
| socket () | Служит для создания сокета в ОС. Имеет три параметра. Первый указывает, к какому семейству протоколов относится создаваемый сокет, а второй и третий параметры определяют конкретный протокол внутри данного семейства. |
| сonnect () | Служит для установки логического соединения со стороны клиента. Имеет три параметра: дескриптор активного сокета, через который будет устанавливаться соединение, полный адрес сокета сервера и его длина |
| read() и write() | Используются для чтения и записи файлов. Могут быть использованы для обмена данных, если в качестве параметров вместо дескрипторов файлов в них задаются дескрипторы сокетов. |
| bind() | Настраивает адрес для созданного сокета. Первый параметр вызова должен содержать дескриптор сокета, для которого производится настройка адреса. Второй и третий параметры задают этот адрес. |
| listen() | Прослушивает соединения на сокете. Позволяет показать готовность принимать соединения и задать лимит входящих соединений. В качестве первого параметра используется дескриптор сокета. Второй параметр определяет максимальную длину очереди входящих соединений. |
| htons() | преобразует узловой порядок расположения байтов положительного короткого целого hostshort в сетевой порядок расположения байтов. |
| accept() | Эта функция используется сервером для принятия связи на сокет. Сокет должен быть уже слушающим в момент вызова функции. Если сервер устанавливает связь с клиентом, то функция accept возвращает новый сокет-дескриптор, через который и происходит общение клиента с сервером. Пока устанавливается связь клиента с сервером, функция accept блокирует другие запросы связи с данным сервером, а после установления связи "прослушивание" запросов возобновляется. |
| ntohs() | преобразует сетевой порядок расположения байтов положительного короткого целого netshort в узловой порядок расположения байтов. |

**2.5 Пример работы приложения**

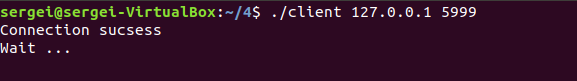
Сначала надо запустить программу сервер, введя в командной строке:

./server [port].

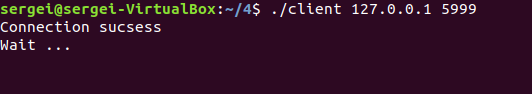
Затем надо запустить программу клиент, для которой формат ввода данных имеет следующий вид:

./client [ip] [port]

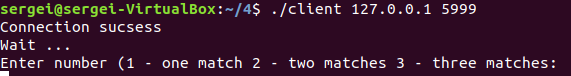
Результат, выведенный на экран 1 игрока:



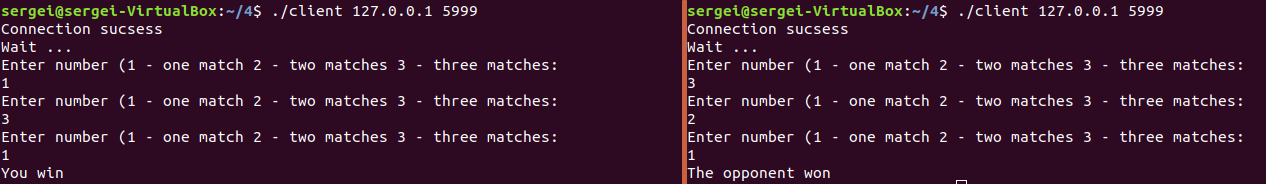
Результат, выведенный на экран 2 игрока:



Теперь выбранный игрок должен начать игру:



Игра продолжается до тех пор, пока один из игроков не заберет последний предмет:



**2.6 Результат работы**

В ходе выполнения данной курсовой работы была создана работающая сетевая игра «Одиннадцать предметов», поддерживающая операционной системы семейства Unix/Linux, в которой для сетевого взаимодействия игроков используются клиент-серверная модель, а обмен данными осуществляется через TCP – сокеты.

**ЗАКЛЮЧЕНИЕ**

Таким образом, в результате курсовой работы мы создали серверное приложение. Приложение работает на операционных системах семейства Linux. Для построения сетевой части приложения использован язык C и компилятор gcc. Сетевое взаимодействие было построено с помощью протокола TCP.

**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Брюс Моли «Unix/Linux. Теория и практика программирования»
2. Карпов В. Е, Коньков К. А «Основы операционных систем. Курс лекций» - Учебное пособие, 2005 - 536с.
3. Скотт Граннеман «Linux. Карманный справочник.»

Листинг В1 — Код из файла server.с

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/ipc.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <time.h>

void game(int listenfd, int cl1, int cl2)

{

int number = 11;

srand(time(0));

int count = number + rand() % number;

while (1)

{

int check;

for (int i = count; i > 0; i--)

{

if (i % 2 == 0)

{

check = 0;

check = htons(check);

write(cl1, &check, sizeof(int));

read(cl1, &check, sizeof(int));

check = ntohs(check);

if (check == 1)

{

number--;

if (number <= 0)

{

check = 1;

check = htons(check);

write(cl1, &check, sizeof(int));

check = 2;

check = htons(check);

write(cl2, &check, sizeof(int));

break;

}

}

else if (check == 2)

{

number = number - 2;

if (number <= 0)

{

check = 1;

check = htons(check);

write(cl1, &check, sizeof(int));

check = 2;

check = htons(check);

write(cl2, &check, sizeof(int));

break;

}

}

else

{

number = number - 3;

if (number <= 0)

{

check = 1;

check = htons(check);

write(cl1, &check, sizeof(int));

check = 2;

check = htons(check);

write(cl2, &check, sizeof(int));

break;

}

}

}

else

{

check = 0;

check = htons(check);

write(cl2, &check, sizeof(int));

read(cl2, &check, sizeof(int));

check = ntohs(check);

if (check == 1)

{

number--;

if (number <= 0)

{

check = 1;

check = htons(check);

write(cl2, &check, sizeof(int));

check = 2;

check = htons(check);

write(cl1, &check, sizeof(int));

break;

}

}

else if (check == 2)

{

number = number - 2;

if (number <= 0)

{

check = 1;

check = htons(check);

write(cl2, &check, sizeof(int));

check = 2;

check = htons(check);

write(cl1, &check, sizeof(int));

break;

}

}

else

{

number = number - 3;

if (number <= 0)

{

check = 1;

check = htons(check);

write(cl2, &check, sizeof(int));

check = 2;

check = htons(check);

write(cl1, &check, sizeof(int));

break;

}

}

}

}

}

}

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

{

int listenfd = 0;

struct sockaddr\_in serveraddr;

char buf[2049];

listenfd = socket(AF\_INET, SOCK\_STREAM, 0);

memset(&serveraddr, '0', sizeof(serveraddr));

memset(buf, '0', sizeof(buf));

serveraddr.sin\_family = AF\_INET;

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

serveraddr.sin\_port = htons(atoi(argv[1]));

bind(listenfd, (struct sockaddr \*)&serveraddr, sizeof(serveraddr));

listen(listenfd, 10);

int cl1 = accept(listenfd, (struct sockaddr \*)NULL, NULL);

int cl2 = accept(listenfd, (struct sockaddr \*)NULL, NULL);

game(listenfd, cl1, cl2);

}

Листинг В2 — Код из файла client.с

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/ipc.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <time.h>

void game(int sockfd)

{

while (1)

{

int check;

read(sockfd, &check, sizeof(int));

check = ntohs(check);

if (check == 0)

{

printf("Enter number (1 - one match 2 - two matches 3 - three matches: \n");

do

{

scanf("%d", &check);

} while(check != 1 && check != 2 && check != 3);

check = htons(check);

write(sockfd, &check, sizeof(int));

}

if (check == 1)

{

printf("You win\n");

break;

}

if (check == 2)

{

printf("The opponent won\n");

break;

}

}

}

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

{

int sockfd = 0, n = 0;

char buf[2049];

struct sockaddr\_in servaddr;

memset(buf, '0', sizeof(buf));

if ((sockfd = socket(AF\_INET, SOCK\_STREAM, 0)) < 0)

{

printf("\n socket creation failed \n");

return 1;

}

memset(&servaddr, '0', sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(atoi(argv[2]));

inet\_pton(AF\_INET, argv[1], &servaddr.sin\_addr);

if (connect(sockfd, (struct sockaddr \*)&servaddr, sizeof(servaddr)) < 0)

{

printf("\nConnect failed.\n");

return 1;

}

printf("Connection sucsess\n");

printf("Wait ...\n");

game(sockfd);

}