**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

Практические приемы построения многопоточных приложений. Вариант 19

**Исполнитель**

Студентка группы БПИ195

Логунова Г.Д.

Задание

У одной очень привлекательной студентки есть N поклонников. Традиционно в день св. Валентина очень привлекательная студентка проводит романтический вечер с одним из поклонников. Счастливый избранник заранее не известен. С утра очень привлекательная студентка получает N «валентинок» с различными вариантами романтического вечера. Выбрав наиболее заманчивое предложение, студентка извещает счастливчика о своем согласии, а остальных – об отказе. Требуется создать многопоточное приложение, моделирующее поведение студентки. При решении использовать парадигму «клиент-сервер» с активным ожиданием.

Составление​​ программы

http://eos.ibi.spb.ru/umk/5\_8/5/5\_R0\_T6.html

Клиенты постоянно кидают запрос на сервер, спрашивая приняла ли решения девушка и если да, то кого выбрала. Это реализовано в функции client. Также есть поток server, с которым клиенты общаются через глобальную переменную decision. Server сообщает клиентам, что решение все еще непринято, если значение переменной decision = -1, в противном случае сообщает клиентам номер выбранной валентинки.

Текст программы:

#include <iostream>

#include <omp.h>

using namespace std;

int desicion = -1;

// вариант 19

void server(int N) {

int check\_desicion;

do {

#pragma omp critical

{

cout << "Введите № выбранной валентинки: ";

}

cin >> check\_desicion;

cout << endl;

if (check\_desicion > N) {

cout << "Номер выбранной валентинки не может превышать общее количесвто валентинок: " << endl;

}

} while (check\_desicion > N);

desicion = check\_desicion;

}

void client(int my\_number) {

while (desicion == -1) {

continue;

}

if (desicion != my\_number) {

#pragma omp critical

{

cout << "Валентинка №" << my\_number << " отклонена" << endl;

}

}

else {

#pragma omp critical

{

cout << "валентинка №" << my\_number << " принята" << endl;

}

}

}

int main()

{

setlocale(LC\_ALL, "rus");

int N;

cout << "Введите количество валентинок N: ";

cin >> N;

server(N);

#pragma omp parallel for

for (int i = 0; i < N; i++) {

client(i);

}

}