ГОУ ВПО «Омский технический университет»

Кафедра Информатики и вычислительной техники

Специальность 230100 «Информатика и вычислительная техника»

Курсовая работа

по дисциплине «Операционные системы»

Вариант 62

Выполнил:

_____

Группа: В-223

Проверил:

Флоренсов А.Н.

_____

Омск, 2015

Задание:

Разработать программную модель ямской почтовой связи, имевшейся в Российской империи. Модель представляет 8 почтовых станций, между которыми перемещаются почтовые подводы. За каждой почтовой станцией закреплено некоторое число лошадей. (Для модели выбрать число в пределах 5-7). Это количество может быть не полностью готовым для перевозок (лошади отдыхают после гоньбы), промоделировать случайными величинами, случайным образом формируются лица, едущие по казенной надобности, случайным образом в один из городов. Граф дорожной связи городов связный, линейный контур, замкнутый или нет - по выбору разработчика. Ввести 2 приоритета для пользующихся услугами ямской связи, более приоритетные едут в 2 раза быстрее. Подобрать характеристики программных генераторов ездоков, чтобы на станциях возникали ситуации ожидания освободившихся или отдохнувших лошадей. Для упрощения модели в подводу для путешественника впрягается одна лошадь, для высокоприоритетных - две. Разработку провести в ОС типа Linux как многопоточную программную имитацию в консольном окне. Поведение модели должно отображаться с помощью символов по усмотрению разработчика. Поведение каждого пассажира должно реализовываться отдельной нитью. Для правильного взаимодействия использовать семафоры или мьютексы.

Описание программы:

Семафор — объект, ограничивающий количество потоков, которые могут войти в заданный участок кода. Семафор — это объект, с которым можно выполнить три операции. Увеличить, уменьшить счетчик, ожидание пока не станет больше 0.

Вот некоторые из проблем, которые могут решать семафоры:

- запрет одновременного выполнения заданных участков кода;
- поочерёдный доступ к критическому ресурсу (важному ресурсу, для которого невозможен (или нежелателен) одновременный доступ).

В данной программе, поочередной доступ к критическому ресурсу не нужен.

POSIX Threads — стандарт POSIX реализации потоков (нитей) выполнения. Стандарт POSIX.1c, Threads extensions (IEEE Std 1003.1c-1995) определяет API для управления потоками, их синхронизации и планирования.

Программа строится из несколько этапов разработки:

1. Постороение схемы почтовой связи.
2. Написане программы, реализующую данную задачу.
3. Тестирование на правильность выполнения.

Первый этап - это этап разработки алгоритмов. Это самый важный этап, так как от него зависит правильность состовления и выполнения исходного кода программы.

Для начало нам надо описать личьности которые будут использоваться у нас в программе.

1. Люди, которые пришли на станцыю за казенной надобностью (Men).
2. Лошади, как средстрво перевозки людей(Loshads).

К этим данным можно отнести и такую сушьность как город-станцию (gorods), в которой и происходят все события.

После описания всех сушьностей можно перейти к разработке алгоритном данной задачи. И как всегда стоит начать с описания того или иного алгоритма.

Для данной задачи было выделенно несколько блоков, которые в свою очередь приведут к правельному решению задачи. Таких блоков несколько:

1. Настройка программы(Main()).
2. Описание прихода людей на ту или иную станцыю(thread_Men()).
3. Добавление заказа(add()).

4. Удаление заказа(deletee()).
5. Отправка лошади в из пункта А в пункт Б(loshad()).
6. Графическое представление бега лошади(beg()).
7. Вывод данных на экран консоли(Writer()).

Сначала программа запускает потоки лошадей и людей, пареллельно им запускается поток «Вывода данных» он позволяет вывести на экран монитора данные лошадей и людей. В программе используется ряд констант, которые по желанию разработчика можно изменить, таких констант в программе три:

#define Ng 8 — Количество городов-станцый.

#define Nl 30 — Количество лошадей

#define Nm 40 — Количество людей

Они служат для решения нашей задачи. Также в программе есть несколько очень важных переменних:

struct lass loshads[Nl]; — Массив лошадей.

int iterator[Ng]; — Массив людей.

sem_t losha[Ng]; — Семафор.

Что бы было топятно как все работает, ниже представлены схемы алгоритмов

Схема №1

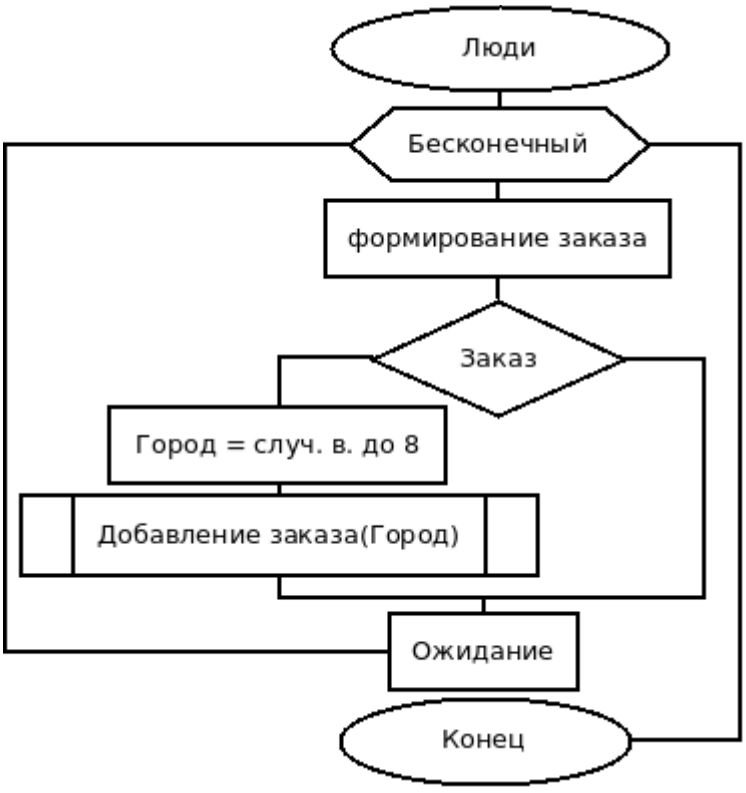

Схема №2

Схема №3

Схема №4

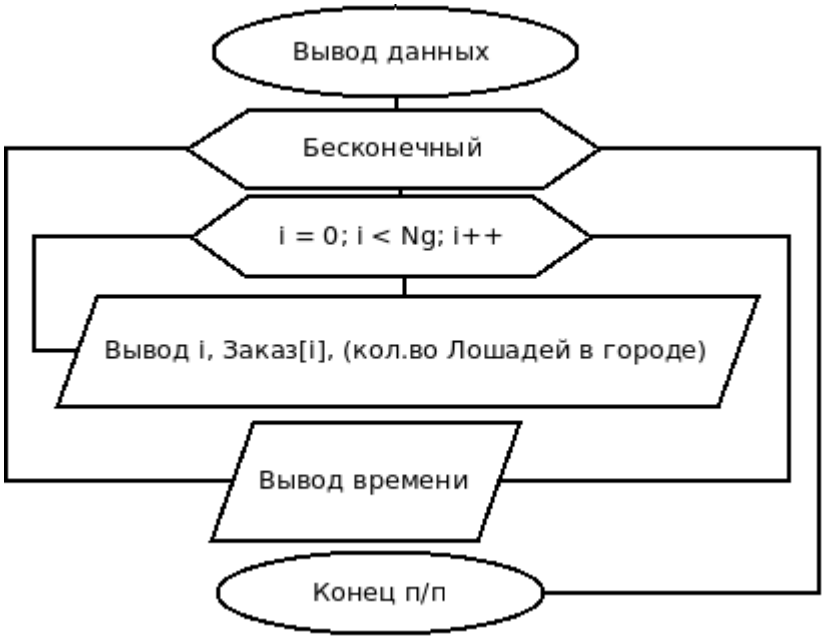

Схема №5

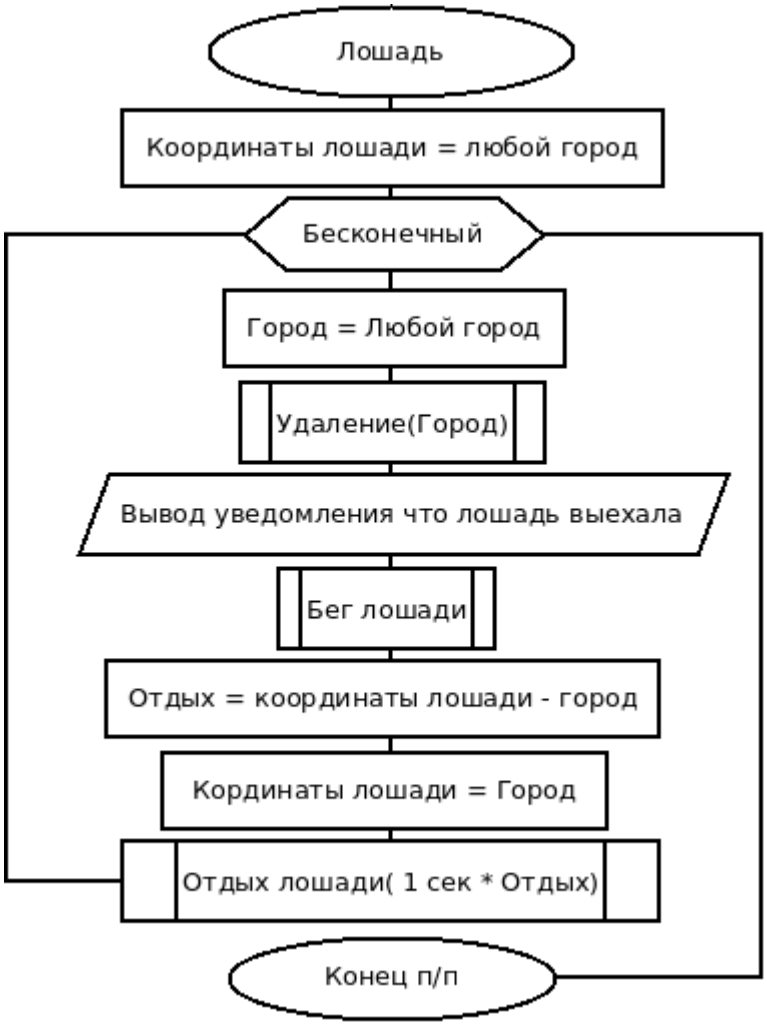

Схема №6

Так как мы имеем БОЛЬШОЙ опыт работы в сфере разработки приложений, это дает нам возможность, сразу представить тот или иной черный яшик, как код.

Код программы:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <pthread.h>
#include <unistd.h>
#include <semaphore.h>
#define Ng 8
#define Nl 30
#define Nm  40
struct lass{
 int x,y;
}loshads[Nl];

int iterator[Ng];
sem_t gorodaAs,losha[9];
void add(int n){
    sem_wait(&losha[n]);
    iterator[n]++;
    sem_post(&losha[n]);
}
void deletee(int n){
    if(iterator[n]==0) return;
    sem_wait(&losha[n]);
    iterator[n]--;
    sem_post(&losha[n]);
}
void *thread_Men(void *in){
   int e= (int)in;
   for(;;){
    int r = rand()%100;
    if (r>10){
        add(rand()%8);
    }
```

```
    usleep(9000000);
    }
}
int lenloshad(int n){
int t,ret=0;
    for (t=0;t<Nl;t++) if (loshads[t].x==n) ret++;
     return ret;
}
void Writer(){
int i,k=0;
    for(;;){
         printf("\033[0;0HНомер города");
        for(i=0; i<Ng;i++){
            printf("\033[%d;0H%d) %2d людей ожидает, свободных лощадей %3d ",i+2,i+1, iterator[i]
,lenloshad(i) );
        }
        usleep(1000000);
        k++;
        printf("\033[24;0H Время: %4d секунд", k);
    }
}
void beg(int l,int l2){
int u = l2 - l;
int i,j,y;
if(u<0) y = 1;
else y = -1;
for(i=0;i<8;i++){
        printf("\033[%d;%dH&",l+2,45+i);
        usleep(100000);
        printf("\033[%d;%dH  ",l+2,45+i);
    }
    for(i=8;i>0;i--){
        printf("\033[%d;%dH&",l+2,45+i);
        usleep(100000);
        printf("\033[%d;%dH  ",l+2,45+i);
```

```
    }
}
void *loshad(void *in){
 int n = (int) in;
 int t,t2;
 loshads[n].x = rand()%Ng;
 usleep(20000);
 for (;;){
        t = rand()%Ng;
           deletee(loshads[n].x);
           printf("\033[23;0HЛощадь выехала из A(%d) в пункт B(%d).. ",loshads[n].x,t);
           beg( loshads[n].x ,t);
           t2 = abs(loshads[n].x-t);
           loshads[n].x =t ;
      usleep((3000000)*t2);
 }
}
int main()
{
int i;
    pthread_t cd[101];
    setbuf(stdout,NULL);
    for(i=0;i<Nl;i++)
    {
        pthread_create( &cd[i], NULL, loshad, (void*)i);
        sem_init(&losha[i],0,3);
    }
           pthread_create( &cd[100], NULL, Writer, NULL);
    for (i=0;i<Nm;i++)
        pthread_create( &cd[Nl+i], NULL, thread_Men, NULL);
        while(1){}
    getchar();
}
```

Список литературы

[1]Флоренсов А.Н. Операционные системы для программиста. Учеб.пос.-Омск,Изд-во ОмГТУ,2005.-240 с.