**МИНОБРНАУКИ РОССИИ**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ  
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»  
(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

РГЗ

дисциплина: Математическая логика и теория алгоритмов  
тема: «Машина Поста»

Выполнил: ст. группы ВТ-  
  
Проверил: Бондаренко Т.В

Белгород 20

**Машина Поста** – это абстрактная (несуществующая реально) вычислительная машина, созданная для уточнения (формализации) понятия алгоритма. Представляет собой универсальный исполнитель, позволяющий вводить начальные данные и читать результат выполнения программы.

Машина Поста состоит из …

1. **бесконечной ленты**, поделенной на одинаковые ячейки (секции). Ячейка может быть пустой (0 или пустота) или содержать метку (1 или любой другой знак),
2. **головки (каретки)**, способной передвигаться по ленте на одну ячейку в ту или иную сторону, а также способной проверять наличие метки, стирать и записывать метку.

Всего для машины Поста существует шесть типов команд:

* **V j** - поставить метку, перейти к j-й строке программы.
* **X j** - стереть метку, перейти к j-й строке программы.
* **<- j** - сдвинуться влево, перейти к j-й строке программы.
* **-> j** - сдвинуться вправо, перейти к j-й строке программы.
* **? j1; j2** - если в ячейке нет метки, то перейти к j1-й строке программы, иначе перейти к j2-й строке программы.
* **!** – конец программы (стоп).

У команды «стоп» отсылки нет.

**Код программы**

#include **<stdio.h>**#include **<stdlib.h>**#define **COMMAND** 10  
**short** arr[3000];  
**char** command1[**COMMAND**];*//команда***int** command2[**COMMAND**] = { -1, -1, -1, -1, -1, -1, -1, -1, -1, -1 };*//куда идёт***int** command3[**COMMAND**] = { -1, -1, -1, -1, -1, -1, -1, -1, -1, -1 };*//куда идёт  
  
  
  
//вывод ленты***void** output() {  
 printf(**"\n"**);  
 **short** \*point = arr + 1490;  
 **for** (**int** i = 0; i < 20; i++) {  
 **if** (\*point)  
 printf(**"V"**);  
 **else** printf(**"X"**);  
 point++;  
 }  
 getchar();  
 getchar();  
}  
*//Вывод списка команд***void** OutputCommand(**int** num) {  
 **for** (**int** i = 0; i < **COMMAND**; i++) {  
 **if** (i == num)  
 printf(**">>"**);  
 **if** (command1[i] == **'?'**)  
 printf(**"%d.\'%c\',%d %d\n"**, i+1,command1[i], command2[i] + 1, command3[i] + 1);  
 **else** printf(**"%d.\'%c\',%d\n"**, i+1, command1[i], command2[i] + 1);  
 }  
}  
  
*//чтение команды***void** read(**int** num) {  
 **int** path;  
 printf(**"Enter command: "**);  
 getchar();  
 command1[num] = getchar();  
 printf(**"Enter next:"**);  
 scanf(**"%d"**, &path);  
 command2[num] = path - 1;  
 **if** (command1[num] == **'?'**) {  
 printf(**"Enter next: "**);  
 scanf(**"%d"**, &path);  
 command3[num] = path - 1;  
 }  
}  
  
*//очистка команды***void** clean(**int** num) {  
  
 command1[num] = 0;  
 command2[num] = 0;  
 command3[num] = 0;  
  
}  
  
**void** start() {  
 **short** \*point = arr + 1500;  
 **int** i = 0;  
 **int** flag = 1;  
 **while** (flag) {  
 **switch** (command1[i]) {  
 **case**(**'v'**):  
 \*point = 1;  
 i = command2[i];  
 **break**;  
 **case**(**'x'**):  
 \*point = 0;  
 i = command2[i];  
 **break**;  
 **case**(**'>'**):  
 point += 1;  
 i = command2[i];  
 **break**;  
 **case**(**'<'**):  
 point -= 1;  
 i = command2[i];  
 **break**;  
 **case**(**'?'**):  
 **if** (\*point)  
 i = command3[i];  
 **else** i = command2[i];  
 **break**;  
 **case**(**'!'**):  
 printf(**"End"**);  
 flag = 0;  
 **break**;  
 **default**:  
 printf(**"Error command"**);  
 }  
  
 }  
 output();  
}  
  
  
*//Интерфейс машины Поста***int** Interface() {  
 **int** num = 0;  
 **char** c;  
 **int** flag = 1;  
 **while** (flag) {  
 OutputCommand(num);  
 printf(**"\nw-Up\na-Clean\ns-Down\nd-Read\n+-Start\n!-Exit\nCommand:"**);  
 c = getchar();  
 **switch** (c) {  
 **case**(**'w'**):  
 num -= 1;  
 **break**;  
 **case**(**'a'**):  
 clean(num);  
 **break**;  
 **case**(**'s'**):  
 num += 1;  
 **break**;  
 **case**(**'d'**):  
 read(num);  
 **break**;  
 **case**(**'+'**):  
 start();  
 **break**;  
 **case**(**'!'**):  
 flag = 0;  
 **break**;  
 **default**:  
 printf(**"Error command\n"**);  
 }  
 system(**"cls"**);  
 }  
 **return** 1;  
}  
  
**int** main() {  
  
 Interface();  
 getchar();  
 **return** 0;  
}