МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

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

Лабораторная работа №4

по дисциплине: Математическая логика и теория алгоритмов

Тема: «Теория алгоритмов»

Выполнил: ст. группы ПВ-211
Чувилко Илья Романович

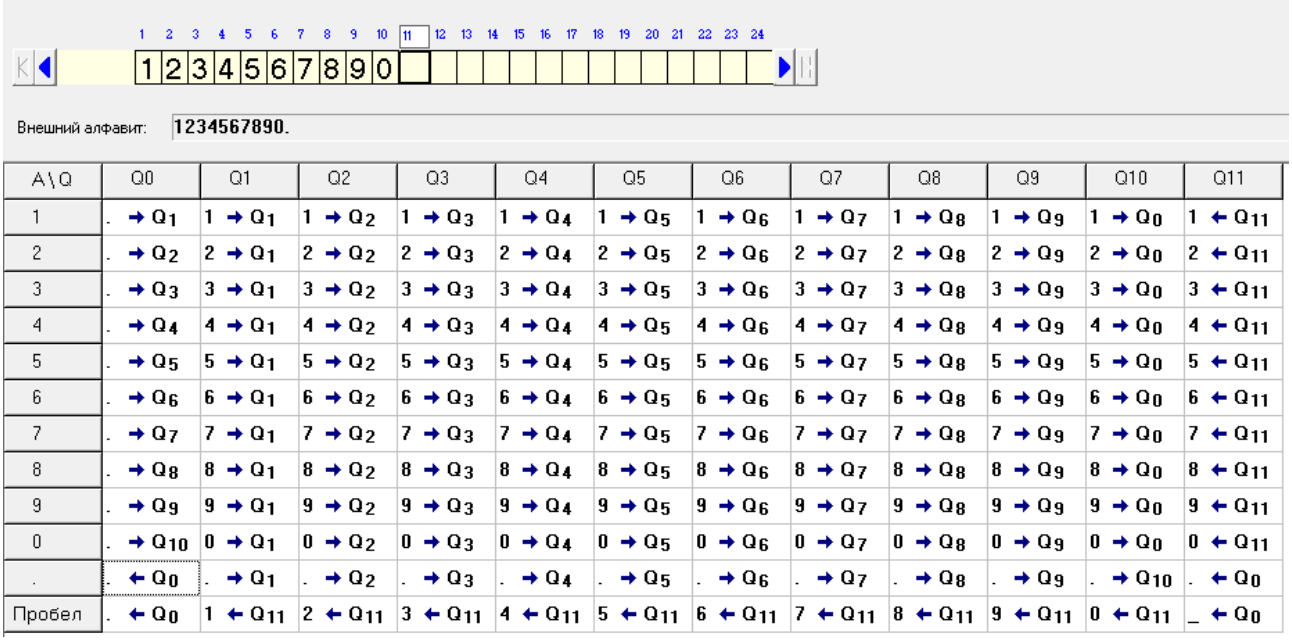
Проверили:
Куценко Дмитрий Александрович

Белгород 2022 г.

Вариант 23

9. На ленте машины Тьюринга записано число в десятичной системе счисления. Головка находится над крайней правой цифрой. Записать цифры этого числа в обратном порядке.

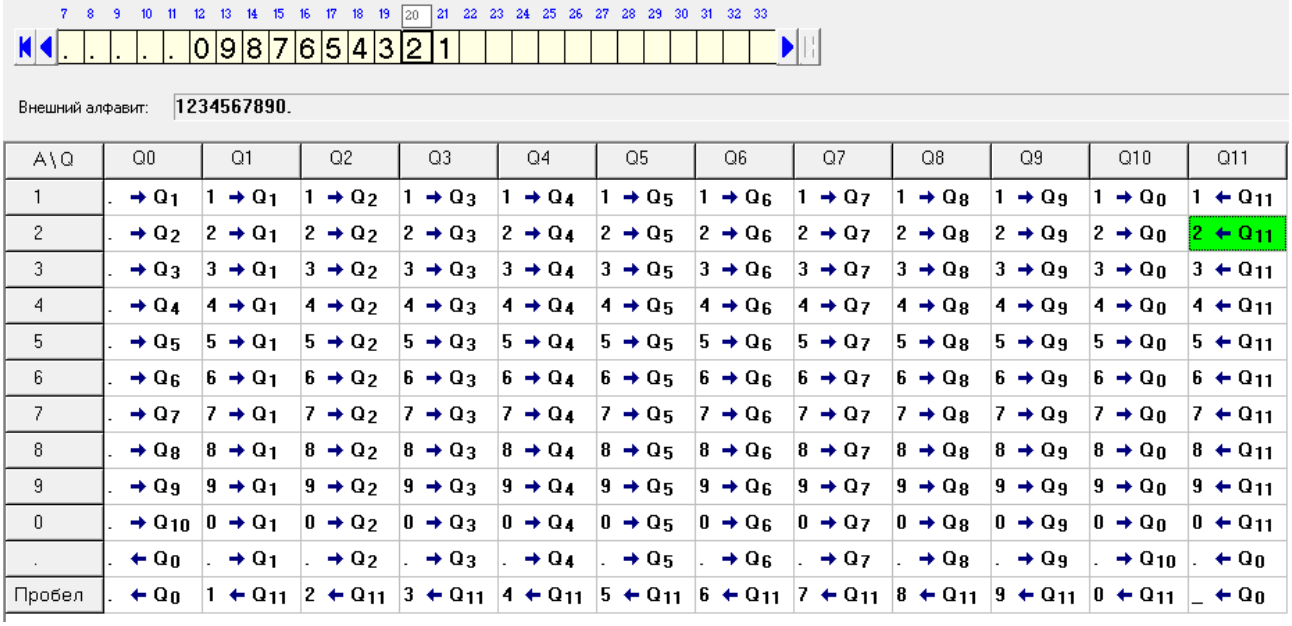
Исходное положение в эмуляторе Algo2000



Внешний алфавит: 1234567890.

A\Q	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11
1	. → Q1	1 → Q1	1 → Q2	1 → Q3	1 → Q4	1 → Q5	1 → Q6	1 → Q7	1 → Q8	1 → Q9	1 → Q0	1 ← Q11
2	. → Q2	2 → Q1	2 → Q2	2 → Q3	2 → Q4	2 → Q5	2 → Q6	2 → Q7	2 → Q8	2 → Q9	2 → Q0	2 ← Q11
3	. → Q3	3 → Q1	3 → Q2	3 → Q3	3 → Q4	3 → Q5	3 → Q6	3 → Q7	3 → Q8	3 → Q9	3 → Q0	3 ← Q11
4	. → Q4	4 → Q1	4 → Q2	4 → Q3	4 → Q4	4 → Q5	4 → Q6	4 → Q7	4 → Q8	4 → Q9	4 → Q0	4 ← Q11
5	. → Q5	5 → Q1	5 → Q2	5 → Q3	5 → Q4	5 → Q5	5 → Q6	5 → Q7	5 → Q8	5 → Q9	5 → Q0	5 ← Q11
6	. → Q6	6 → Q1	6 → Q2	6 → Q3	6 → Q4	6 → Q5	6 → Q6	6 → Q7	6 → Q8	6 → Q9	6 → Q0	6 ← Q11
7	. → Q7	7 → Q1	7 → Q2	7 → Q3	7 → Q4	7 → Q5	7 → Q6	7 → Q7	7 → Q8	7 → Q9	7 → Q0	7 ← Q11
8	. → Q8	8 → Q1	8 → Q2	8 → Q3	8 → Q4	8 → Q5	8 → Q6	8 → Q7	8 → Q8	8 → Q9	8 → Q0	8 ← Q11
9	. → Q9	9 → Q1	9 → Q2	9 → Q3	9 → Q4	9 → Q5	9 → Q6	9 → Q7	9 → Q8	9 → Q9	9 → Q0	9 ← Q11
0	. → Q10	0 → Q1	0 → Q2	0 → Q3	0 → Q4	0 → Q5	0 → Q6	0 → Q7	0 → Q8	0 → Q9	0 → Q0	0 ← Q11
.	. ← Q0	. → Q1	. → Q2	. → Q3	. → Q4	. → Q5	. → Q6	. → Q7	. → Q8	. → Q9	. → Q10	. ← Q0
Пробел	. ← Q0	1 ← Q11	2 ← Q11	3 ← Q11	4 ← Q11	5 ← Q11	6 ← Q11	7 ← Q11	8 ← Q11	9 ← Q11	0 ← Q11	_ ← Q0

Итоговое положение в эмуляторе Algo2000

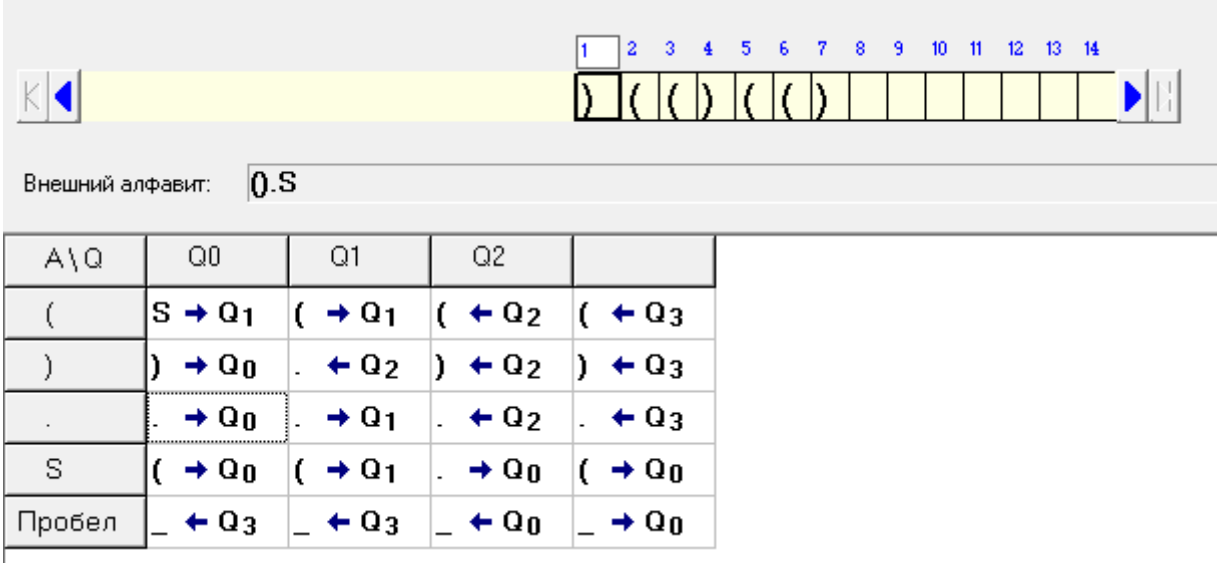


Внешний алфавит: 1234567890.

A\Q	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11
1	. → Q1	1 → Q1	1 → Q2	1 → Q3	1 → Q4	1 → Q5	1 → Q6	1 → Q7	1 → Q8	1 → Q9	1 → Q0	1 ← Q11
2	. → Q2	2 → Q1	2 → Q2	2 → Q3	2 → Q4	2 → Q5	2 → Q6	2 → Q7	2 → Q8	2 → Q9	2 → Q0	2 ← Q11
3	. → Q3	3 → Q1	3 → Q2	3 → Q3	3 → Q4	3 → Q5	3 → Q6	3 → Q7	3 → Q8	3 → Q9	3 → Q0	3 ← Q11
4	. → Q4	4 → Q1	4 → Q2	4 → Q3	4 → Q4	4 → Q5	4 → Q6	4 → Q7	4 → Q8	4 → Q9	4 → Q0	4 ← Q11
5	. → Q5	5 → Q1	5 → Q2	5 → Q3	5 → Q4	5 → Q5	5 → Q6	5 → Q7	5 → Q8	5 → Q9	5 → Q0	5 ← Q11
6	. → Q6	6 → Q1	6 → Q2	6 → Q3	6 → Q4	6 → Q5	6 → Q6	6 → Q7	6 → Q8	6 → Q9	6 → Q0	6 ← Q11
7	. → Q7	7 → Q1	7 → Q2	7 → Q3	7 → Q4	7 → Q5	7 → Q6	7 → Q7	7 → Q8	7 → Q9	7 → Q0	7 ← Q11
8	. → Q8	8 → Q1	8 → Q2	8 → Q3	8 → Q4	8 → Q5	8 → Q6	8 → Q7	8 → Q8	8 → Q9	8 → Q0	8 ← Q11
9	. → Q9	9 → Q1	9 → Q2	9 → Q3	9 → Q4	9 → Q5	9 → Q6	9 → Q7	9 → Q8	9 → Q9	9 → Q0	9 ← Q11
0	. → Q10	0 → Q1	0 → Q2	0 → Q3	0 → Q4	0 → Q5	0 → Q6	0 → Q7	0 → Q8	0 → Q9	0 → Q0	0 ← Q11
.	. ← Q0	. → Q1	. → Q2	. → Q3	. → Q4	. → Q5	. → Q6	. → Q7	. → Q8	. → Q9	. → Q10	. ← Q0
Пробел	. ← Q0	1 ← Q11	2 ← Q11	3 ← Q11	4 ← Q11	5 ← Q11	6 ← Q11	7 ← Q11	8 ← Q11	9 ← Q11	0 ← Q11	_ ← Q0

25. Дан массив из открывающихся и закрывающихся скобок. Построить машину Тьюринга, которая удаляла бы пары взаимных скобок, ставя на их месте символы «.» (здесь и далее без кавычек). Например, если дано слово «)(()(()», то его надо преобразовать в слово «)...((.».

Исходное положение в эмуляторе Algo2000



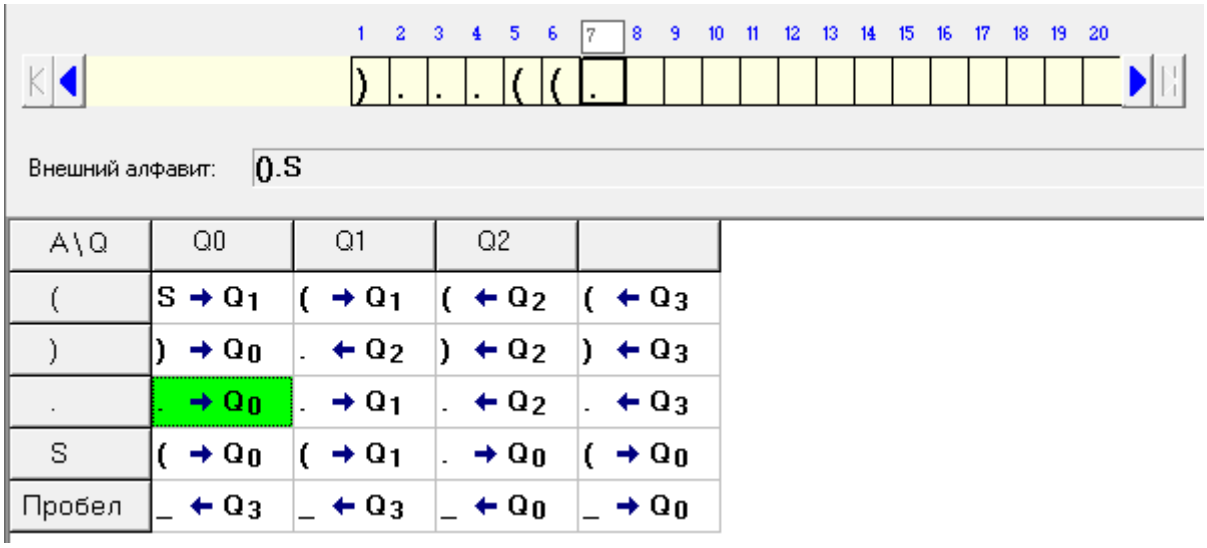
Q0 – Состояние поиска первой открывающейся скобки, которая будет заменена символом S.

Q1 – Состояние, когда одна открывающаяся скобка была найдена и ищется закрывающаяся.

Q2 – Состояние, когда закрывающаяся скобка была найдена. Тогда указатель движется влево до тех пор пока не найдет временный символ S, который он заменит на точку.

Q3 – Состояние, когда закрывающаяся скобка не была найдена. Тогда указатель движется влево до тех пор пока не найдет временный символ S, который он заменит на открывающуюся скобку.

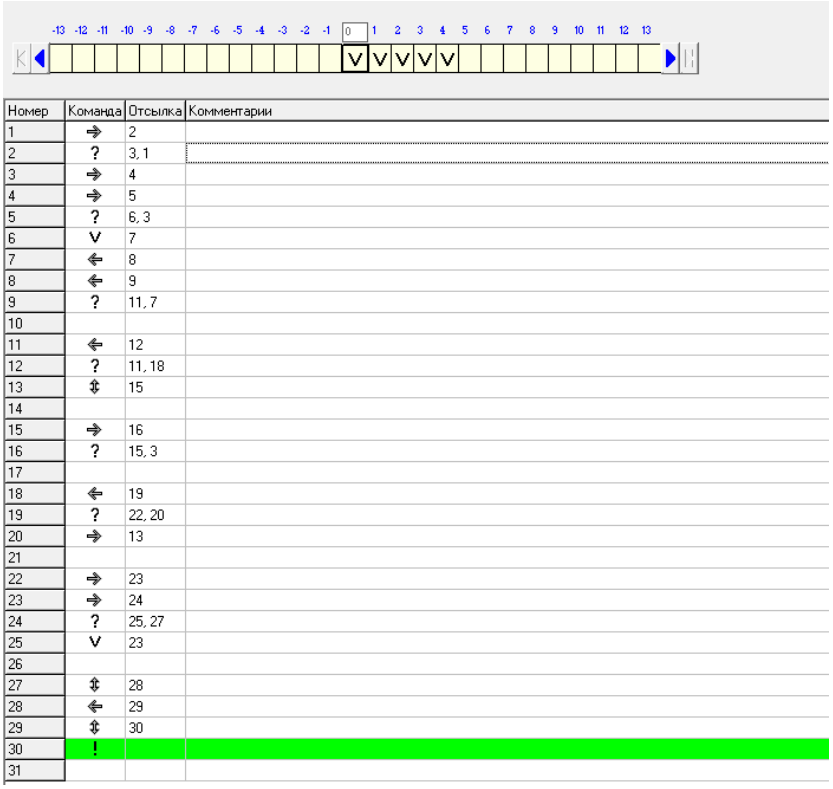
Итоговое положение в эмуляторе Algo2000



62. Число k представляется на ленте машины Поста k + 1 идущими подряд

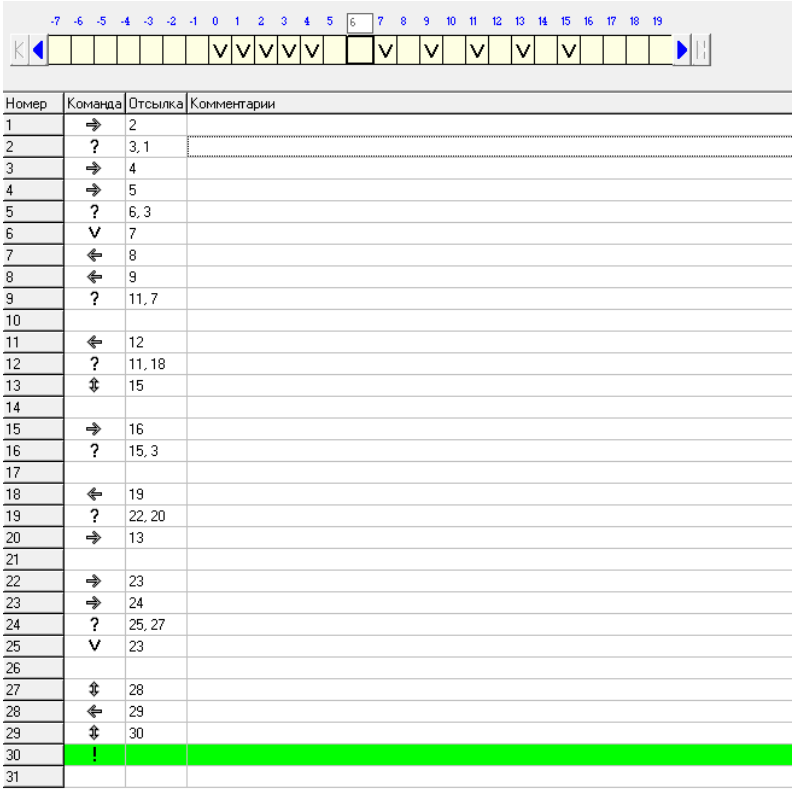
метками. Одна метка соответствует нулю. Составить программу копирования исходного числа так, чтобы во втором числе метки располагались на ленте через одну пустую ячейку. Головка расположена над одной из меток, принадлежащих заданному числу. При этом исходное число должно остаться на ленте.

Исходное положение в эмуляторе Algo2000



Номер	Команда	Отсылка	Комментарии
1	⇒	2	
2	?	3, 1	
3	⇒	4	
4	⇒	5	
5	?	6, 3	
6	V	7	
7	⇐	8	
8	⇐	9	
9	?	11, 7	
10			
11	⇐	12	
12	?	11, 18	
13	⇕	15	
14			
15	⇒	16	
16	?	15, 3	
17			
18	⇐	19	
19	?	22, 20	
20	⇒	13	
21			
22	⇒	23	
23	⇒	24	
24	?	25, 27	
25	V	23	
26			
27	⇕	28	
28	⇐	29	
29	⇕	30	
30	!		
31			

Итоговое положение в эмуляторе Algo2000



Номер	Команда	Отсылка	Комментарии
1	⇒	2	
2	?	3, 1	
3	⇒	4	
4	⇒	5	
5	?	6, 3	
6	V	7	
7	⇐	8	
8	⇐	9	
9	?	11, 7	
10			
11	⇐	12	
12	?	11, 18	
13	⇕	15	
14			
15	⇒	16	
16	?	15, 3	
17			
18	⇐	19	
19	?	22, 20	
20	⇒	13	
21			
22	⇒	23	
23	⇒	24	
24	?	25, 27	
25	V	23	
26			
27	⇕	28	
28	⇐	29	
29	⇕	30	
30	!		
31			

№68.4 Доказать, что следующие функции примитивно рекурсивны:

N68.4

$$Sg(x) = \begin{cases} 0 & x = 0 \\ 1 & x > 1 \end{cases}$$

Применим оператор примитивной рекурсии:

$$\begin{cases} Sg(0) = O(x) \\ Sg(x+1) = C \end{cases}$$

подставим C = 1, получим:

$$Sg(x) = \begin{cases} 0, & x = 0 \\ 1, & x > 0 \end{cases}$$

=> примитивно рекурсивна. ч.т.д.

Практическая часть:

Разработать программу на языке Brainfuck, решающую задачу «На информационной ленте машины Тьюринга в трех ячейках в произвольном порядке записаны три различные буквы “A” (здесь и далее без кавычек), “B” и “C”. Необходимо составить функциональную схему машины Тьюринга, которая сумеет поменять местами крайние буквы»

Пошаговая реализация на языке Brainfuck

- Считываем. Ввод заканчиваем 0.
  >,[>,] < [−]

- Идем в начало.
  < [<] >

- Копируем первую букву в соседнюю ячейку, исходную букву удаляем.
  [− < + >]

- Идем в конец и копируем последнюю букву в первую.
  >> [− << + >>]

- Идем в самое начало и копируем содержимое в последнюю ячейку.
  <<< [− >>> + <<<]

- Вывод результата.
  > [. >]

Команда Brainfuck	Эквивалент на Си	Описание команды
Начало программы	`int i = 0;` `char arr[30000];` `memset(arr, 0, sizeof(arr));`	выделяется память под 30 000 ячеек с нулевыми начальными значениями
`>`	`i++;`	перейти к следующей ячейке
`<`	`i--;`	перейти к предыдущей ячейке
`+`	`arr[i]++;`	увеличить значение в текущей ячейке на 1
`-`	`arr[i]--;`	уменьшить значение в текущей ячейке на 1
`.`	`putchar(arr[i]);`	напечатать значение из текущей ячейки
`,`	`arr[i] = getchar();`	ввести извне значение и сохранить в текущей ячейке
`[`	`while(arr[i]){`	если значение текущей ячейки ноль, перейти вперёд по тексту программы на ячейку, следующую за соответствующей `]` (с учётом вложенности)
`]`	`}`	если значение текущей ячейки не нуль, перейти назад по тексту программы на символ `[` (с учётом вложенности)

Реализация интерпретатора на языке C

```
#include <iostream>
#include <windows.h>
#include <ctime>

void read_bfFile(FILE *cmdlist) {
  char c;
  int i = 0;
  while (!feof(cmdlist)) {
    if (i == 50) {
      printf("\n");
      i = 0;
    }
    fscanf(cmdlist, "%c", &c);
    if (!feof(cmdlist)) {
      printf("%c", c);
      i++;
    }
  }
  printf("\n");
}

void translate_brainfuck_fFile(FILE *cmdlist, char *tier) {
  int loop;
  char cmd_char;

  do {
    fscanf(cmdlist, "%c", &cmd_char);
    switch (cmd_char) {
      case '>':
        tier++;
        break;
      case '<':
        tier--;
        break;
      case '+':
        (*tier)++;
        break;
      case '-':
        (*tier)--;
        break;
      case '.':
        putchar(*tier);
```

```
      break;
    case ',':
      *tier = getchar();
      break;
    case '[':
      if (!*tier) {
        loop = 1;
        while (loop) {
          fscanf(cmdlist, "%c", &cmd_char);
          if (cmd_char == '[') {
            loop++;
          }
          if (cmd_char == ']') {
            loop--;
          }
        }
      }
      break;
    case ']':
      if (*tier) {
        loop = 1;
        while (loop) {
          fseek(cmdlist, -2, SEEK_CUR);
          fscanf(cmdlist, "%c", &cmd_char);
          if (cmd_char == '[') {
            loop--;
          }
          if (cmd_char == ']') {
            loop++;
          }
        }
      }
      break;
    default:
      fprintf(stderr, "Error");
      exit(1);
  }
  (*tier)++;
 } while ((!feof(cmdlist)));
 printf("\n");
}

int main() {
 SetConsoleOutputCP(CP_UTF8);
 srand(time(NULL));

 char cpu[30000];
 FILE *read = fopen("exercise.txt", "r");
 read_bfFile(read);
 translate_brainfuck_fFile(read, cpu);
}
```

Вывод: В ходе лабораторной работы изучили на машину Тьюринга и машину Поста, а так же реализовали интерпретатор языка Brainfuck. По результатам работы, сделали вывод, что современные языки программирования куда более просты для понимания и реализации алгоритмов.