УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе № 1

по предмету «Компиляторные технологии»

Вариант 1

Выполнили:

Габрусь С. П., Панкратьев Е. С.

гр. 251003

Проверил:

Шостак Е. В.

Минск 2023

# Постановка задачи

Для заданного вариантом вида текстовых данных проанализировать допустимые значения и разработать:

* регулярное выражение**;**
* НКАдля этого регулярного выражения;
* эквивалентный ДКА **и его** таблицу переходов**;**
* программное средство**,** реализующее работу этого ДКА.

Программное средство должно поддерживать следующие возможности:

* проверка корректности произвольной строки;
* поиск всех подстрок, соответствующих требованиям, в произвольной строке.

# Регулярное выражение

# НКА

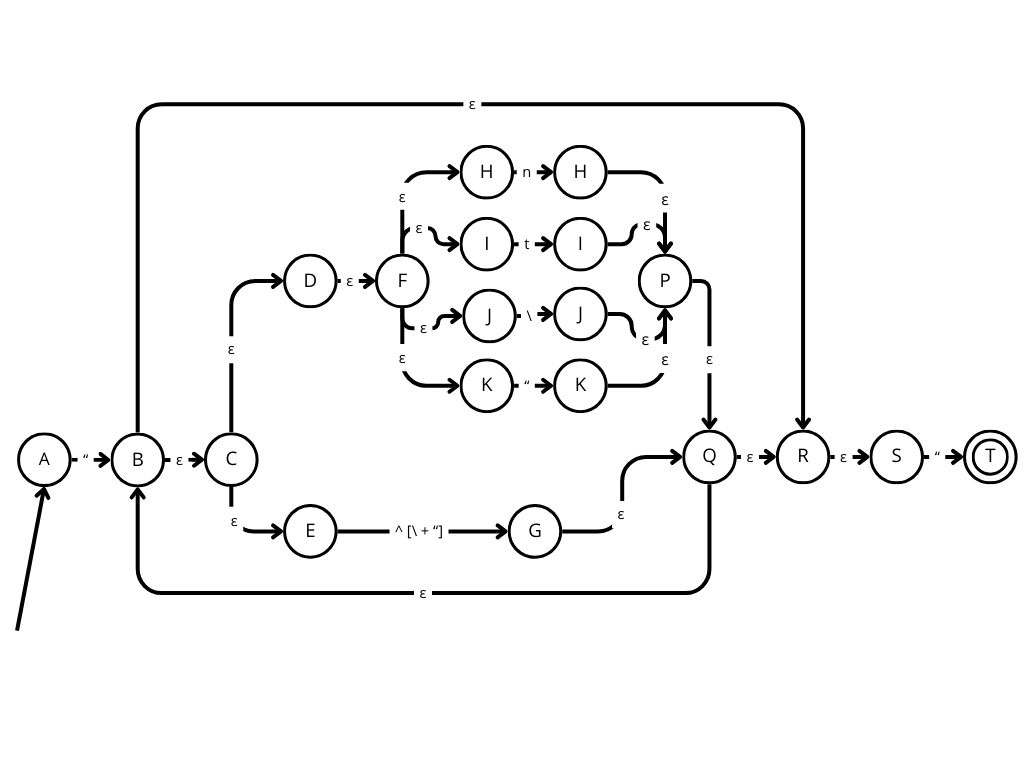


Рисунок 1 – НКА

# ДКА

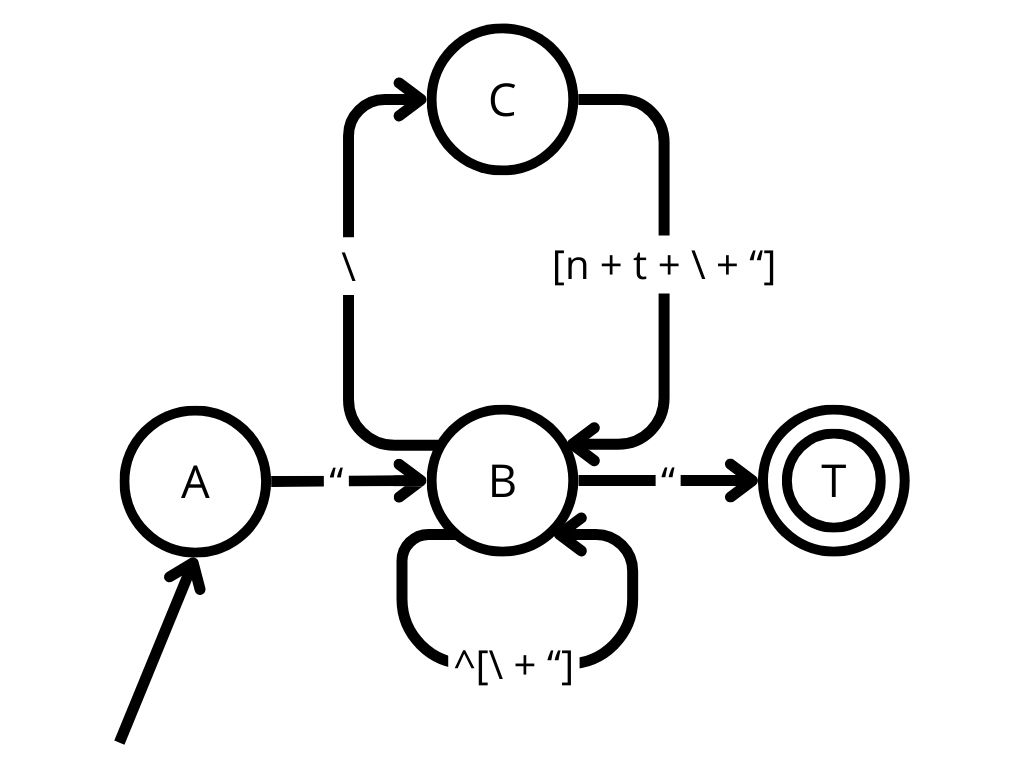


Рисунок 2 – ДКА

# Таблица переходов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Текущее состояние | " | \ | n, t | ^ [" + \] |
| A | A | - | - | - |
| B | C | D | - | B |
| C | - | - | - | - |
| D | D | B | B | - |

Приложение А

(обязательное)

Исходный код программы

#include <stdio.h>  
  
typedef enum {  
 **ERROR** = -1,  
 **A** = 0,  
 **B**,  
 **C**,  
 **D**,  
} State;  
  
const State Transitions[4][4] = {  
*// " \ n, t, ^ [" + \]* { **B**, **ERROR**, **ERROR**, **ERROR** }, *// A* { **C**, **D**, **ERROR**, **B** }, *// B* { **ERROR**, **ERROR**, **ERROR**, **ERROR** }, *// C* { **B**, **B**, **B**, **ERROR** } *// D*};  
  
int GetType(const char symbol) {  
 switch (symbol) {  
 case '"':  
 return 0;  
 case '\\':  
 return 1;  
 case 'n':  
 case 't':  
 return 2;  
 default:  
 return 3;  
 }  
}  
  
int CheckSubStringLiteral(const char \*str, const int size) {  
 State state = **A**;  
 for (int i = 0; i < size; ++i) {  
 state = Transitions[state][GetType(str[i])];  
 if (state == **ERROR**)  
 return 0;  
 }  
 return state == **C**;  
}  
  
void PrintSubStringValidity(const char \*str, const int size) {  
 if (CheckSubStringLiteral(str, size)) {  
 printf("%.\*s\n", size, str);  
 printf("--------------------------------------------------------------------------------------------------\n");  
 }  
}  
  
void PrintValidStringLiterals(const char \*str) {  
 printf("Valid substrings:\n");  
 for (int i = 0; str[i] != '\n'; ++i) {  
 for (int j = i + 1; str[j] != '\n'; ++j)  
 PrintSubStringValidity(&str[i], j - i + 1);  
 }  
}  
  
void clearInputBuffer() {  
 int c;  
 while ((c = getchar()) != '\n' && c != EOF);  
}  
int needToContinue() {  
 int choice;  
 printf("Do you want to continue? Enter 1 to continue.\n");  
 scanf("%d", &choice);  
 clearInputBuffer();  
 return choice == 1;  
}  
  
int main() {  
 do {  
 printf("Enter a string:\n");  
 char str[256];  
 if (fgets(str, sizeof(str), stdin) != NULL)  
 PrintValidStringLiterals(str);  
 else  
 return -1;  
 } while(needToContinue());  
  
 return 0;  
}