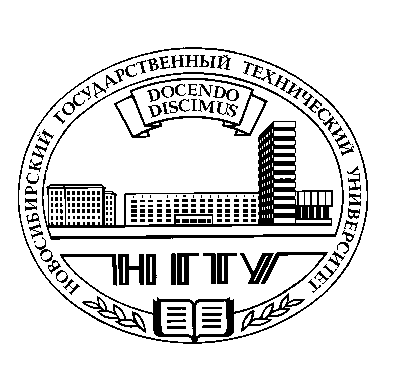
**Министерство науки, высшей школы и технической политики Российской Федерации.**

**Новосибирский Государственный Технический Университет.**



**Курсовая работа по системному программированию.**

Синтаксический распознаватель арифметического оператора условного перехода языка FORTRAN.

**Факультет:** АВТ.

**Кафедра:** АСУ.

**Группа:** А–513.

**Студент:** Бойко Константин Анатольевич.

**Преподаватель:** Шорников Юрий Владимирович.

**Ассистент:** Панова Вера Борисовна.

**Дата:** 30 апреля 1997 г.

**Отметка о защите:**

Новосибирск.

**Язык оператора.**

**Язык арифметического оператора условного перехода языка FORTRAN.**

**IF(<АВ>) LABEL1,LABEL2,LABEL3**

**АВ -** Арифметическое Выражение.

**LABEL1,LABEL2,LABEL3 -** первая, вторая и третья метки

соответственно.

**Грамматика языка.**

G[<Оператор>]:

1. <Оператор> → IF(<АВ>)<LABEL>,<LABEL>,<LABEL>
2. <LABEL> → <ЦБЗ>
3. <АВ> → T | <АВ>T | <АВ>-T
4. T → O | T\*O | T/O | T\*\*O
5. O → (<АВ>) | <Идентификатор> | <ЧПЗ>
6. <Идентификатор> → Б{Б|Ц}
7. <ЦБЗ> → Ц{Ц}[Λ]
8. <ЧПЗ> → Ц{Ц}[.Ц{Ц}]

Условные обозначения:

|  |  |  |
| --- | --- | --- |
| **Т** | **–** | **ТЕРМ.** |
| **О** | **–** | **ОПЕРАНД.** |
| **Б** | **–** | **БУКВА.** |
| **Ц** | **–** | **ЦИФРА.** |
| **ЦБЗ** | **–** | **ЦЕЛОЕ БЕЗ ЗНАКА.** |
| **\*\*** | **–** | **ВОЗВЕДЕНИЕ В СТЕПЕНЬ.** |
| **LABEL** | **–** | **МЕТКА.** |
| **ЧПЗ** | **–** | **ЧИСЛО С ПЛАВАЮЩЕЙ ЗАПЯТОЙ.** |
| **Λ** | **–** | **КОНЕЦ СТРОКИ (пусто).** |
| **АВ** | **–** | **АРИФМЕТИЧЕСКОЕ ВЫРАЖЕНИЕ.** |

**Классификация грамматики.**

Данная грамматика G[<ОПЕРАТОР>], согласно классификации Хомского, является контекстно-свободной, так как правая часть каждой редукции начинается либо с терминального символа, либо с нетерминального, принадлежащего объединённому словарю.

**A → a, A∈Vn, a∈V\*.**

Грамматика G[<ОПЕРАТОР>] не является автоматной, так как не все её редукции начинаются с терминального символа. По этой же причине данная грамматика не является S - грамматикой.

**Метод анализа.**

Для данной грамматики реализован разбор методом рекурсивного спуска, поскольку она относится к классу контекстно-свободных.

Идея метода состоит в том, что каждому нетерминальному символу ставится в соответствие определённая программная единица (функция), которая распознаёт цепочку, порождаемую этим нетерминалом.

Эти процедуры и функции вызываются в соответствии с правилами грамматики и иногда вызывают сами себя.

Данный метод реализован на языке C++, поскольку он обладает рекурсивными возможностями.

**Диагностика и нейтрализация ошибок.**

Для данной грамматики производится только диагностика и нейтрализация ошибок. Исправление ошибок не производится.

Нейтрализация ошибок осуществляется по методу Айронса, то есть, спускаясь по синтаксическому дереву без возврата по контексту, при обнаружении тупиковой ситуации отбрасываются те литеры (символы), которые привели в тупиковую ситуацию и разбор продолжается.

**Тестирование.**

⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

IF(((6/t+g\*456)/(9.347\*y-555))\*\*6) 64,44,548

⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

Текущий символ ( - 40.

AB - Проверка на Арифметическое Выражение.

Текущий символ ( - 40.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

AB - Проверка на Арифметическое Выражение.

Текущий символ ( - 40.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

AB - Проверка на Арифметическое Выражение.

Текущий символ 6 - 54.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа 6.

FLOAT - проверка на число с плавающей запятой с цифры 6.

Текущий символ / - 47.

Текущий символ t - 116.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа t.

Текущий символ + - 43.

AB - Проверка на Арифметическое Выражение.

Текущий символ g - 103.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа g.

Текущий символ \* - 42.

Текущий символ 4 - 52.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа 4.

FLOAT - проверка на число с плавающей запятой с цифры 4.

Текущий символ 5 - 53.

Текущий символ 6 - 54.

Текущий символ ) - 41.

Текущий символ / - 47.

Текущий символ ( - 40.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

AB - Проверка на Арифметическое Выражение.

Текущий символ 9 - 57.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа 9.

FLOAT - проверка на число с плавающей запятой с цифры 9.

Текущий символ . - 46.

Текущий символ 3 - 51.

Текущий символ 4 - 52.

Текущий символ 7 - 55.

Текущий символ \* - 42.

Текущий символ y - 121.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа y.

Текущий символ - - 45.

AB - Проверка на Арифметическое Выражение.

Текущий символ 5 - 53.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа 5.

FLOAT - проверка на число с плавающей запятой с цифры 5.

Текущий символ 5 - 53.

Текущий символ 5 - 53.

Текущий символ ) - 41.

Текущий символ ) - 41.

Текущий символ \* - 42.

Текущий символ \* - 42.

Текущий символ 6 - 54.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа 6.

FLOAT - проверка на число с плавающей запятой с цифры 6.

Текущий символ ) - 41.

Текущий символ - 32.

Текущий символ 6 - 54.

ABS - проверка на Целое Без Знака с цифры 6.

Текущий символ 4 - 52.

Текущий символ , - 44.

Текущий символ 4 - 52.

ABS - проверка на Целое Без Знака с цифры 4.

Текущий символ 4 - 52.

Текущий символ , - 44.

Текущий символ 5 - 53.

ABS - проверка на Целое Без Знака с цифры 5.

Текущий символ 4 - 52.

Текущий символ 8 - 56.

Текущий символ NULL - 0.

⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

IF(r\*\*34/50.22) 3,56,1

⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

Текущий символ ( - 40.

AB - Проверка на Арифметическое Выражение.

Текущий символ r - 114.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа r.

Текущий символ \* - 42.

Текущий символ \* - 42.

Текущий символ 3 - 51.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа 3.

FLOAT - проверка на число с плавающей запятой с цифры 3.

Текущий символ 4 - 52.

Текущий символ / - 47.

Текущий символ 5 - 53.

Term - Проверка на Терм.

Operand - Проверка на Операнд.

IDENT - Проверка на Идентификатор с символа 5.

FLOAT - проверка на число с плавающей запятой с цифры 5.

Текущий символ 0 - 48.

Текущий символ . - 46.

Текущий символ 2 - 50.

Текущий символ 2 - 50.

Текущий символ ) - 41.

Текущий символ - 32.

Текущий символ 3 - 51.

ABS - проверка на Целое Без Знака с цифры 3.

Текущий символ , - 44.

Текущий символ 5 - 53.

ABS - проверка на Целое Без Знака с цифры 5.

Текущий символ 6 - 54.

Текущий символ , - 44.

Текущий символ 1 - 49.

ABS - проверка на Целое Без Знака с цифры 1.

Текущий символ NULL - 0.

⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

**Листинг программы.**

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// FILE "KURSOVIC.CPP".

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// ВАРИАHТ # 2.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Арифметический оператор условного перехода языка FORTRAN.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Кафедpа : АСУ.

// Гpуппа : А-513.

// Студент : Бойко Константин Анатольевич.

// Пpеподаватели : кандидат технических наук, доцент Шоpников Юpий Владимиpович,

// ассистент Панова Веpа Боpисовна.

// Дата : 30 апреля 1997г.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Заголовочные файлы.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

# include<stdio.h>

# include<conio.h>

# include<string.h>

# include<stdlib.h>

# include "keyboard.h"

# include<dos.h>

# include <ctype.h>

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Константы.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

# define FILE\_NAME "TEXT.TXT"

# define YES 1

# define NO 2

# define OK 3

//# define TEST

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Глобальные переменные.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

char file1[22][60]; // Массив строк.

char mes[22][30][50]; // Массив сообщений об ошибках.

char bufer[4096]; // Снимок с экрана.

int com=0; // Показатель : откомпилирован файл или нет.

int pos=2; // Текущая позиция в строке.

char string[80]; // Текущая строка файла.

char nx; // Текущий символ в строке.

int e1=0; // Счетчик страниц массива собщений об ошибках.

int e2=0; // Счетчик строк в странице.

int l=0; // Счетчик меток.

FILE\* fl=fopen("MESSAGE.TXT","wt"); // Файл с сообщениями об ошибках.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Прототипы функций.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Windows(int,int,int,int,char\*); // Построение окон.

void Beep(int,int); // Звуковой сигнал.

void Open\_File(void); // Открытие файла.

void Second\_Window(void); // Построение нижней панели.

void MyPuts(char\*,int); // Аналог puts(char\*).

void Print\_File(void); // Печать файла.

void Menu(void); // Меню.

int i\_readkey(void); // Опрос клавиатуры.

void Help(void); // Помощь.

void Grammar(void); // Грамматика языка.

void Language(void); // Язык оператора.

void Compile(void); // Компиляция.

void Operator(char\*); // Сканирование оператора.

void Scan(void); // Сканирование следующего символа.

void Label(void); // Реализация Label.

int ABS(void); // Реализация ABS.

int IDENT(void); // Реализация IDENT.

int AB(void); // Реализация AB.

void Error(int); // Обработка ошибки.

int Term(void); // Реализация Term.

int Operand(void); // Реализация Operand.

int Float(void); // Реализация Float.

void Message(int); // Вывод сообщений об ошибках.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Функция MAIN.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void main()

{

\_setcursortype(\_SOLIDCURSOR);

window(1,1,80,25);

textattr(113);

clrscr();

for(int i=0;i<24;i++)

cputs("");

textcolor(BLACK);

cputs(" Enter filename");

window(20,7,60,17);

textattr(127);

clrscr();

Windows(20,7,60,17,"Open a File");

gotoxy(4,3);

puts("Name ");

gotoxy(6,7);

puts("По умолчанию будет введен файл");

gotoxy(17,8);

puts(FILE\_NAME);

window(23,10,57,10);

textattr(31);

clrscr();

Open\_File();

Second\_Window();

Print\_File();

\_setcursortype(\_NOCURSOR);

Menu();

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Звуковой сигнал.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Beep(int tone,int pause)

{

sound(tone);

delay(pause);

nosound();

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Построение рамок.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Windows(int x,int y,int x1,int y1,char\* string)

{

y1++;

for(int i=2;i<x1-x;i++)

{

gotoxy(i,1);

putch('=');

gotoxy(i,y1-y);

putch('=');

}

for(i=1;i<y1-y;i++)

{

gotoxy(2,i);

putch('¦');

gotoxy(x1-x,i);

putch('¦');

}

gotoxy(2,1);

putch('г');

gotoxy(x1-x,1);

putch('¬');

gotoxy(x1-x,y1-y);

putch('-');

gotoxy(2,y1-y);

putch('L');

gotoxy((x1-x-strlen(string)+2)/2,1);

puts(string);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Функция открытия файла.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Open\_File(void)

{

char filename[38];

filename[0]=35;

char\* p=cgets(filename);

strcpy(filename,p);

if(filename[0]==NULL)

strcpy(filename,FILE\_NAME);

FILE \*file;

if ((file = fopen(filename, "rt"))== NULL)

{

window(1,1,80,25);

textattr(7);

\_setcursortype(\_NORMALCURSOR);

clrscr();

perror("ERROR");

fcloseall();

getch();

exit(1);

}

for(int i=0;i<22;i++)

{

char temp[255];

if((fgets(temp,80,file))==NULL)

break;

strcpy(file1[i],temp);

}

fclose(file);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Построение нижней панели.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Second\_Window(void)

{

window(1,1,80,25);

textattr(17);

clrscr();

window(1,1,80,24);

textattr(31);

clrscr();

Windows(1,1,80,24,"File");

window(1,25,80,25);

textattr(127);

clrscr();

MyPuts(" ~F1~ Help ~F2~ Grammar ~F3~ Language ~F4~ Compile ~ESC~ Exit",

LIGHTBLUE);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// MyPuts.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void MyPuts(char\* string,int color)

{

int j=0;

for(int i=0;string[i] != NULL;i++)

{

if(string[i]=='~' && j==0)

{

textcolor(color);

i++;

j++;

}

if(string[i]=='~' && j!=0)

{

textcolor(BLACK);

i++;

j--;

}

cprintf("%c",string[i]);

if(string[i]==NULL)

break;

}

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Функция вывода файла на экран.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Print\_File(void)

{

window(3,2,77,24);

textattr(31);

for(int i=0;i<22;i++)

{

cputs(file1[i]);

putch('\r');

}

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Меню.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Menu(void)

{

gettext(1,1,80,25,bufer); // Вся информация с экрана заносится в буфер.

int y=1; // Счетчик строк.

gotoxy(1,1);

textattr(63);

clreol();

cputs(file1[0]);

textattr(31);

gotoxy(5,23);

cprintf(" %-2d ",y);

while(1)

{

switch (i\_readkey())

{

case EASC\_F1:

gettext(1,1,80,25,bufer);

Help();

puttext(1,1,80,25,bufer);

break;

case EASC\_F2:

gettext(1,1,80,25,bufer);

Grammar();

puttext(1,1,80,25,bufer);

break;

case EASC\_F3:

gettext(1,1,80,25,bufer);

Language();

puttext(1,1,80,25,bufer);

break;

case EASC\_F4:

gettext(1,1,80,25,bufer);

Compile();

puttext(1,1,80,25,bufer);

window(48,2,78,23);

textattr(95);

clrscr();

Windows(49,3,79,24,"Message");

Message(y-1);

break;

case EASC\_Esc:

window(1,1,80,25);

textattr(7);

clrscr();

fcloseall();

exit(1);

break;

case EASC\_Up:

if(y>1)

{

if(com>0)

window(3,2,47,24);

else

window(3,2,77,24);

y--;

gotoxy(1,y);

textattr(63);

clreol();

cputs(file1[y-1]);

gotoxy(1,y+1);

textattr(31);

clreol();

cputs(file1[y]);

gotoxy(5,23);

cprintf(" %d ",y);

if(com>0)

Message(y-1);

}

break;

case EASC\_Down:

if(y<22)

{

if(com>0)

window(3,2,47,24);

else

window(3,2,77,24);

gotoxy(1,++y);

textattr(63);

clreol();

cputs(file1[y-1]);

gotoxy(1,y-1);

textattr(31);

clreol();

cputs(file1[y-2]);

gotoxy(5,23);

cprintf(" %d ",y);

if(com>0)

Message(y-1);

}

break;

}

}

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Help.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Help(void)

{

window(1,1,80,25);

textattr(78);

clrscr();

Windows(1,1,80,25,"Help");

window(4,3,77,23);

cprintf("НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ\n\n\r");

cprintf(" КАФЕДРА АСУ\n\n\r");

cprintf(" Курсовая работа по системному программированию на тему:\n\r");

cprintf(" \"Арифметический оператор условного перехода языка FORTRAN\".\n\n\r");

cprintf(" Студента группы А-513\n\n\r");

cprintf(" Бойко Константина Анатольевича.\n\n\r");

cprintf("В программе использованы клавиши:\n\r");

cprintf(" F1 - данный Help.\n\r");

cprintf(" F2 - грамматика языка.\n\r");

cprintf(" F3 - язык оператора.\n\r");

cprintf(" F4 - компиляция файла.\n\r");

cprintf(" ESC - выход из программы.\n\n\r");

cprintf(" Для выхода из Help нажмите любую клавишу.");

getch();

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Грамматика.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Grammar(void)

{

window(10,2,70,23);

textattr(46);

clrscr();

Windows(10,2,70,23,"Grammar");

window(12,4,68,22);

cprintf(" 1. <Оператор> -> IF(<АВ>)<LABEL>,<LABEL>,<LABEL>\n\r");

cprintf(" 2. <LABEL> -> <ЦБЗ>\n\r");

cprintf(" 3. <АВ> -> T | <АВ>+T | <АВ>-T\n\r");

cprintf(" 4. T -> O | T\*O | T/O | T\*\*O\n\r");

cprintf(" 5. O -> (<АВ>) | <Идентификатор> | <ЧПЗ>\n\r");

cprintf(" 6. <Идентификатор> -> Б{Б|Ц}\n\r");

cprintf(" 7. <ЦБЗ> -> Ц{Ц}\n\r");

cprintf(" 8. <ЧПЗ> -> Ц{Ц}[.Ц{Ц}]\n\n\r");

cprintf(" Условные обозначения:\n\r");

cprintf(" T - терм.\n\r");

cprintf(" O - операнд.\n\r");

cprintf(" Б - буква.\n\r");

cprintf(" Ц - цифра.\n\r");

cprintf(" ЦБЗ - целое без знака.\n\r");

cprintf(" ЧПЗ - число с плавающей запятой.\n\r");

cprintf(" \*\* - возведение в степень.\n\r");

cprintf(" <LABEL> - метка.\n\r");

cprintf(" <АВ> - арифметическое выражение.");

getch();

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Язык.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Language(void)

{

window(20,7,60,17);

textattr(62);

clrscr();

Windows(20,7,60,17,"Language");

window(22,9,58,16);

cprintf(" IF(<АВ>) LABEL1,LABEL2,LABEL3\n\n\r");

cprintf(" АВ - Арифметическое Выражение.\n\r");

cprintf(" LABEL1,LABEL2,LABEL3 - первая,вторая и третья метки соответственно.\n\r");

getch();

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Компиляция программы.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Compile(void)

{

window(20,7,55,13);

textattr(94);

clrscr();

Windows(20,7,55,13,"Compile");

window(22,8,58,16);

gotoxy(1,1);

if(com!=0)

{

cprintf("\n\n Файл откомпилирован!");

Beep(900,1000);

return;

}

cputs("\n Идет компиляция файла.\r\n\n Пожалуйста, подождите...");

#ifdef TEST

window(1,1,80,25);

clrscr();

#endif

for(int i=0;file1[i][0]!=NULL;i++)

{

Operator(file1[i]);

if(l!=3 && l!=0)

Error(11);

e1++;

e2=0;

}

com++;

delay(300);

Beep(1000,200);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Оператор.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Operator(char\* str)

{

pos=2;

l=0;

string[0]=NULL;

strcpy(string,str);

string[strlen(string)-1]=NULL;

char temp[3];

temp[0]=str[0];

temp[1]=str[1];

temp[2]=NULL;

#ifdef TEST

puts(string);

puts(temp);

getch();

#endif

fprintf(fl," ⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯\n");

fprintf(fl,"%s\n",string);

if(strcmp(temp,"IF")!=0)

{

#ifdef TEST

printf("Не найден оператор IF.\n");

#endif

Error(10);

return;

}

Scan();

if(nx==' ')

Scan();

if(nx!='(')

Error(7);

if(nx==NULL)

{

Error(12);

Error(9);

return;

}

AB();

while(nx!=')' && nx!=NULL && nx!=' ')

Scan();

if(nx!=')')

Error(3);

if(nx==NULL)

{

Error(9);

return;

}

Scan();

if(nx==NULL)

{

Error(9);

return;

}

if(nx==' ')

Scan();

Label();

if(nx!=',')

{

Error(5);

Scan();

}

else

Scan();

if(nx==NULL)

return;

Label();

if(nx!=',')

{

Error(5);

Scan();

}

else

Scan();

if(nx==NULL)

return;

Label();

if(nx!=NULL)

{

Error(6);

Scan();

}

return;

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Scan.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Scan(void)

{

nx=string[pos++];

#ifdef TEST

printf("Текущий символ %c - %d.\n",nx,nx);

#endif

fprintf(fl,"Текущий символ %c - %d.\n",nx,nx);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Метка.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Label(void)

{

if(ABS() == NO)

{

Error(8);

Scan();

}

else

l++;

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Целое Без Знака.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

int ABS(void)

{

#ifdef TEST

cprintf("ABS - проверка на Целое Без Знака с цифры %c.\r\n",nx);

#endif

fprintf(fl,"ABS - проверка на Целое Без Знака с цифры %c.\n",nx);

if(isdigit(nx))

{

while(isdigit(nx))

Scan();

return(YES);

}

return(NO);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Арифметическое Выражение.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

int AB(void)

{

#ifdef TEST

cprintf("AB - Проверка на Арифметическое Выражение.\r\n");

#endif

fprintf(fl,"AB - Проверка на Арифметическое Выражение.\n");

Scan();

if(nx==')')

{

Error(12);

return(NO);

}

Term();

if(nx=='+')

{

AB();

}

else if(nx=='-')

{

AB();

}

if(nx==NULL)

return(OK);

else if(nx!='\*' && nx!='/' && nx!=')')

Error(1);

return(YES);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Терм.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

int Term(void)

{

#ifdef TEST

cprintf("Term - Проверка на Терм.\r\n");

#endif

fprintf(fl,"Term - Проверка на Терм.\n");

Operand();

if(nx=='/')

{

Scan();

Term();

}

else if(nx=='\*')

{

Scan();

if(nx=='\*')

{

Scan();

Term();

}

else Term();

}

if(nx==NULL)

return(OK);

else if(nx!='+' && nx!='-' && nx!=')')

Error(2);

return(OK);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Операнд.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

int Operand(void)

{

#ifdef TEST

cprintf("Operand - Проверка на Операнд.\r\n");

#endif

fprintf(fl,"Operand - Проверка на Операнд.\n");

if(nx=='(')

{

AB();

if(nx!=')')

Error(3);

else

{

Scan();

return(OK);

}

}

else if(IDENT() == NO)

{

if(Float() == NO)

Error(4);

}

return(OK);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Идентификатор.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

int IDENT(void)

{

#ifdef TEST

cprintf("IDENT - Проверка на Идентификатор с символа %c.\r\n",nx);

#endif

fprintf(fl,"IDENT - Проверка на Идентификатор с символа %c.\n",nx);

if(isalpha(nx))

{

while(isalpha(nx) || isdigit(nx))

Scan();

return(YES);

}

return(NO);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Ошибки.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Error(int num)

{

char \*E[]={

"Ожидается '+' или '-'.",

"Ожидается '\*','/' или '\*\*'.",

"Ожидается ).",

"Ожидается идентификатор или целое без знака.",

"Отсутствует ','.",

"Не конец строки.",

"Ожидается (.",

"Метка - не целое без знака.",

"Отсутствуют метки.",

"Не найден оператор IF.",

"Отсутствует метка.",

"Отсутствует AB.",

NULL

};

sprintf(mes[e1][e2],"%s",E[num-1]);

e2++;

#ifdef TEST

Beep(1000,100);

cprintf("Ошибка: %s\r\n",E[num-1]);

getch();

#endif

fprintf(fl," ⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯\n");

fprintf(fl,"Ошибка: %s\r\n",E[num-1]);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Float.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

int Float(void)

{

#ifdef TEST

cprintf("FLOAT - проверка на число с плавающей запятой с цифры %c.\r\n",nx);

#endif

fprintf(fl,"FLOAT - проверка на число с плавающей запятой с цифры %c.\n",nx);

if(isdigit(nx))

{

while(isdigit(nx))

Scan();

if(nx=='.')

{

Scan();

while(isdigit(nx))

Scan();

}

return(YES);

}

return(NO);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

// Message.

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

void Message(int x)

{

window(50,3,76,22);

textattr(95);

clrscr();

gotoxy(1,1);

for(int i=0;mes[x][i][0]!=NULL;i++)

cprintf("%s\r\n",mes[x][i]);

}

//⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

**Литература.**

1. Курс лекций по системному программированию.
2. Герберт Шилдт «C для профессиональных программистов».