**Программа "текстовый редактор"**

Пояснительная записка к курсовой работе по дисциплине “Основы алгоритмизации и программирования”

Выполнил : студент гр. 96-ВВ3 Курапов А.В.

Пензенский государственный технический университет, кафедра “Вычислительная Техника”

Пенза 1997

**Введение**

С момента появления первых образцов персональных компьютеров прошло не так уж и много времени,но сейчас без них уже немыслимо огромное количество областей человеческой деятельности - экономика , управление , наука , инженерное дело , издательское дело , образование , культура и т.д.

Интерес к персональным компьютерам постоянно растет,а круг их пользователей непрерывно расширяется.В число пользователей ПЭВМ вовлекаются как новички в компьютерном деле,так и специалисты по другим классам ЭВМ.

Язык С++ - универсальный язык общего назначения, область приложений которого - программирование систем в самом широком смысле. Кроме этого, С++ успешно используется как во многих приложениях, так и в мощных операционных системах. Реализация С++ осуществлена для машин в диапазоне от самых простых персональных компьютеров до самых мощных суперкомпьютеров и для всех операционных систем.

**1 Постановка задачи**

Задача курсовой работы состоит в разработки на языке программирования С++ программы текстового редактора , которая должна осуществлять следующие функции:

1) Ввод исходного текста в заданный файл;

2) Вывод содержимого файла на экран;

3) Обработка файла: форматирование текста по заданному образцу методом центрирования.

Парамерты для форматирования:

а) левый отступ;

b) количество символов в строке;

c) красная строка.

4) Поиск заданной в тексте введённой подстроки , после поиска выдать запрос : "продолжить или закончить".

5) Cохранение результатов обработки данных.

Имена фсех файлов задаются пользователем в диалоговом режиме.

**2 Метод решения задачи**

Анализируя постановку задачи становится ясно , что для её решения необходимо применить метод многофункционального программирования - этот метод основан на разбиении программы на отдельные функциональные модули , которые вызываются из основной программы в процессе её работы.

В основе разработки такой программы лежит низходящее программирование - т.е. сначала разрабатывается основная программа - в данном случае программа ввода текста на экран , обработки функциональных клавиш и меню , с помощью выбора пунктов которого будет в дальнейшем организовываться её работа. При выборе того или иного пункта меню или нажатии функциональной клавиши будет вызываться одна или несколько подпрограмм , но на начальном этапе разработки программы методом нисходящего программирования вместо этих подпрограмм ставяться "заглушки" - пустые подпрограммы , которые в процессе разработки будут заменяться программами , выполняющими определённую функцию.

**3 Описание логического представления данных**

Входными данными в программе являются:

Уже существующий текстовый файл - хранится на диске , может быть под любым именем , при загрузки имя файла запрашивается программой в диалоговом режиме , при загрузке имя файла записывается в локальную переменную типа char - fname (25 символов) , уаказатель на начало текста записывается в переменную \*text , а длина текста в переменную len типа int;

Параметры текста (левый отступ , количество символов в строке) - можно изменить в процессе работы программы , начальные значения при загрузки : левый отступ=0 , количество символов=60; Данные хранятся в переменных основной программы - left , right (int).

Текст , вводимый с клавиатуры хранится в переменной text типа char , которая в конце работы может быть сохранена в файле , имя которого задаётся пользователем в диалоговом режиме , имя файла записывается в локальную переменную типа char - fname (25 символов).

Шаблон для поиска , который вводится пользователем с клавиатуры хранится в меременной типа char - mascs длинной в 40 символов

Выходными данными в программе являются:

Текст , вводимый с клавиатуры хранится в переменной text типа char , которая в сохраняется в файле , имя которого задаётся пользователем с клавиатуры в диалоговом режиме , имя файла записывается в локальную переменную типа char - fname (25 символов).

Результат поиска по шаблону - координаты курсора x,y (int) записываются как координаты первого символа найденного по шаблону слова , если слово не найденно , то x=0 , y=0.

6 Спецификация на функциональные модули

6.1 Функция output

|  |  |
| --- | --- |
| Строка вызова: | output(scrs,x,y); |
| Назначение: | Осуществляет вывод на экран текста и строки подсказки и курсора |
| Функции: | Вывод текста , указатель на которого передаётся в строке параметров , вывод нижней строки подсказки , вычисление координат курсора на экране и вывод курсора на экран. |
| Логические  переменные: | \*a (char) - указатель на начало верхней строки  x,ky (int) - позиция курсора в тексте |
| Внутренние  переменные: | np (int) - Количество непечатаемых символов  rx (int) - Реальная координата курсора на экране  y (int) - Координата Y вывода  \* ot (char) - Временный указатель  n (int) - Временная переменная , необходима для вывода строки |
| Вызывает функции: | нет |
| Вызывается из: | основной программы. |

6.2 Функция sinput

|  |  |
| --- | --- |
| Строка вызова: | SInput(fname,20,21,25); - Вызывается из load  SInput(fname,20,21,25); - Вызывается из save  SInput(masks,40,30,25); - Вызывается из основной программы |
| Назначение: | Осуществляет ввод строки. |
| Функции: | Вывод пустой строки , установка координат , вывод строки запроса, считывание строки посимвольно с клавиатуры , выход , если нажата клавиша Enter или Esc , если нажата Esc то строка не записывается. |
| Логические  переменные: | \*s (char) - указатель на строку  lng (int) - максимальная длина  x,y - координаты ввода |
| Внутренние  переменные: | rs[100] (char) - Временная строка  n (int) - Счетчик  c (char) - Введённый с клавиатуры символ |
| Вызывает функции: | нет |
| Вызывается из: | load,save,основной программы. |

6.3 Функция load

|  |  |
| --- | --- |
| Строка вызова: | load(text); - Вызывается из основной программы |
| Назначение: | Осуществляет загрузку текста из файла. |
| Функции: | Запрос имени файла , открытие файла с данным именем , чтение в переменную , указатель которой передаётся в строке параметров , закрытие файла. |
| Логические  переменные: | \*p (char) - указатель на текст |
| Внутренние  переменные: | fname[25] (char) - Имя файла  \*f (file) - указатель на файл |
| Вызывает функции: | sinput |
| Вызывается из: | основной программы. |

6.4 Функция save

|  |  |
| --- | --- |
| Строка вызова: | save(text); - Вызывается из основной программы |
| Назначение: | Осуществляет запись текста в файл. |
| Функции: | Запрос имени файла , открытие файла с данным именем , запись текста , указатель которго передаётся в строке параметров , в файл , закрытие файла. |
| Логические  переменные: | \*p (char) - указатель на текст |
| Внутренние  переменные: | fname[25] (char) - Имя файла  \*f (file) - указатель на файл  \*pr (char) - временный указатель |
| Вызывает функции: | sinput |
| Вызывается из: | основной программы. |

6.5 Функция reset

|  |  |
| --- | --- |
| Строка вызова: | reset(text,32767); - Вызывается из основной программы |
| Назначение: | Осуществляет уничтожение информации. |
| Функции: | Заполнение кодом перевод строки переменной , указанной в сторке параметров , на длинну , указанную в строке параметров. |
| Логические  переменные: | \*a (char) - указатель на текст  n (int) - количество символов в тексте |
| Внутренние  переменные: | m (int) - счётчик |
| Вызывает функции: | нет |
| Вызывается из: | основной программы. |

6.6 Функция up

|  |  |
| --- | --- |
| Строка вызова: | else scrs=up(text,scrs); - Вызывается из основной программы |
| Назначение: | Осуществляет передвижение указателя на одну строку вверх. |
| Функции: | Возвращает указатель на строку , которая выше на одну позицию, чем исходная. |
| Логические  переменные: | \*a (char) - указатель на текст  \*b (char) - указатель на текущую строку |
| Внутренние  переменные: | нет |
| Вызывает функции: | нет |
| Вызывается из: | основной программы. |

6.7 Функция down

|  |  |
| --- | --- |
| Строка вызова: | else scrs=down(text,scrs); - Вызывается из основной программы |
| Назначение: | Осуществляет передвижение указателя на одну строку вниз. |
| Функции: | Возвращает указатель на строку , которая ниже на одну позицию, чем исходная. |
| Логические  переменные: | \*a (char) - указатель на текст  \*b (char) - указатель на текущую строку |
| Внутренние  переменные: | нет |
| Вызывает функции: | нет |
| Вызывается из: | основной программы. |

6.8 Функция menu

|  |  |
| --- | --- |
| Строка вызова: | switch (menu(30,5," Сохранить файл "  " Считать файл "  " Ввод шаблона "  " Параметры "  " Выход из программы "  "\x0"))  Вызывается из основной программы |
| Назначение: | Осуществляет вывод и обработку меню. |
| Функции: | Вывод меню на экран , опрос клавиатуры , при нажатии клавиш иправления курсором (вверх , вниз) перемещение цветового курсора по пунктам меню , выбор определённого пункта меню при нажатии клавиши с номером пункта меню , при нажатии клавиши Enter возвращается номер выбранного пункта меню , при нажатии клавиши Esc возвращается -1. |
| Логические  переменные: | x,y (int) - координаты вывода меню на экран.  \*сapt (char) - указатель на текст меню |
| Внутренние  переменные: | n,m (int) - Счетчики  num (int) - Количество пунктов  k (int) - Выбранный пункт  pt (char) - Временный указатель на символ  c (char) - Считанный с клавиатуры символ |
| Вызывает функции: | нет |
| Вызывается из: | основной программы. |

6.8 Функция add

|  |  |
| --- | --- |
| Строка вызова: | add(st,a,0,0,' '); - Вызывается из centr  add(text,scrs,x,y,10);  add(text,scrs,x,y,13); Вызывается из основной программы  add(text,scrs,x,y,' ');  add(text,scrs,x,y,c); |
| Назначение: | Осуществляет добавление символа в текст. |
| Функции: | Помещение указателя на начало строки , подсчёт символов в строке , вставка символа в строку. |
| Логические  переменные: | \*st (char) - указатель на текст  \*a (char) - указатель на текущую строку  x,y (int) - координаты  c (char) - добавляемай символ |
| Внутренние  переменные: | n (int) - число символов в строке  d (int) - счётчик  \*p (char) - временный указатель |
| Вызывает функции: | нет |
| Вызывается из: | centr , основной программы. |

6.9 Функция end

|  |  |
| --- | --- |
| Строка вызова: | x=end(text,scrs,x,y); - Вызывается из основной программы. |
| Назначение: | Осуществляет поиск конца сироки. |
| Функции: | Возвращает длину строки. |
| Логические  переменные: | \*st (char) - указатель на текст  \*a (char) - указатель на текущую строку  x,y (int) - координаты |
| Внутренние  переменные: | n (int) - число символов в строке  \*p (char) - временный указатель |
| Вызывает функции: | нет |
| Вызывается из: | основной программы. |

6.10 Функция del

|  |  |
| --- | --- |
| Строка вызова: | del(st,a,0,0); - Вызывается из centr.  del(text,scrs,x,y); - Вызывается из основной программы.  del(text,scrs,x-1,y); - Вызывается из основной программы. |
| Назначение: | Осуществляет удаление символа из текста. |
| Функции: | Поиск необходимого места в тексте , удаление символа , если был удалён "перевод строки" и остался "возврат коретки" либо наоюорот , то удаляется ещё один символ. |
| Логические  переменные: | \*st (char) - указатель на текст  \*a (char) - указатель на текущую строку  x,y (int) - координаты |
| Внутренние  переменные: | n (int) - число символов в строке  d (int) - счётчик  \*p (char) - временный указатель |
| Вызывает функции: | нет |
| Вызывается из: | centr , основной программы. |

6.11 Функция find

|  |  |
| --- | --- |
| Строка вызова: | find(text,&scrs,&x,&y,masks); - Вызывается из основной программы. |
| Назначение: | Осуществляет поиск фрагмента в тексте , начиная с текущего положения курсора. |
| Функции: | Поиск текущего места в тексте , поиск фрагмента , если фрагмент найден - перемещение курсора. |
| Логические  переменные: | \*st (char) - указатель на текст  \*scrs (char) - указатель на текущую строку  \*x , \*y (int) - указатель на координаты |
| Внутренние  переменные: | n (int) - число символов в строке  d (int) - счётчик  k (int) - счётчик  \*p (char) - временный указатель |
| Вызывает функции: | нет |
| Вызывается из: | основной программы. |

6.12 Функция dinput

|  |  |
| --- | --- |
| Строка вызова: | left=DInput(30,25); - Вызывается из основной программы.  right=DInput(30,25); - Вызывается из основной программы. |
| Назначение: | Осуществляет ввод числа. |
| Функции: | Вывод строки запроса , считывание цифр с клавиатуры , выход , если нажата клавиша Enter или Esc , если нажата Esc то число не записывается не записывается. Возвращает считанное число в формате long. |
| Логические  переменные: | x , y (int) - координаты ввода |
| Внутренние  переменные: | s[100] (char) - временная строка  rz (long) - cчитанное число  n (int) - счётчик  c (char) - символ |
| Вызывает функции: | нет |
| Вызывается из: | основной программы. |

6.13 Функция centr

|  |  |
| --- | --- |
| Строка вызова: | centr(text,scrs,y,left,right); - Вызывается из основной программы. |
| Назначение: | Осуществляет центровку текущей строки. |
| Функции: | Поиск текущего местоположения , удаление пробелов в начале строки , подсчёт длины строки , определение количесва добавляемых пробелов , добавление пробелов. |
| Логические  переменные: | \*st (char) - указатель на текст  \*scrs (char) - указатель на текущую строку  x , y (int) - координаты  left (int) - левый отступ  right (int) - число символов в строке |
| Внутренние  переменные: | n (int) - число символов в строке  d (int) - счётчик |
| Вызывает функции: | add , del |
| Вызывается из: | основной программы. |

6.14 Основная программа

|  |  |
| --- | --- |
| Назначение: | Осуществляет опрос клавиатуры , редактирование текста , вывод меню на экран , обработку пунктов меню , и функциональных клавиш , вызов функций в соответствии с выбранным действием. |
| Функции: | Выделение памяти под текст , вывод текста на экран , считывание кода нажатой клавиши , анализ полученного кода и вызов соответствующей функции или выполнение действия. |
| Внутренние  переменные: | left (int) - левый отступ  right (int) - число символов в строке  masks (char) - маска для поиска  \*scrs (char) - указатель на текущую строку  x , y (int) - координаты курсора  c (char) - введённый с клавиатуры символ  n (int) - счётчик |
| Вызывает функции: | output , sinput , load , save , reset , up , down , menu , add , end , del , find , dinput , centr . |

**7 Технические требования для использования**

Для запуска данной программы необходимо наличие персонального компьютера IBM PC/XT c операционной системой MS-DOS.

Необходимый объем оперативной памяти - 400 Кбайт.

Дла запуска программы необходимо сделать текущем каталог с программой tr.exe и вызвать её на выполнение.

Программа работает в текстовом режиме 80х25 символов.

Файлы программы , если не указывать дополнительный путь , будут записываться и считываться из текущего каталога текущего диска.

Управление программой усуществляется с клавиатуры.

**8 Описание работы программы**

После своей загрузки программа осуществляет оформление экрана , вывод в нижнюю часть экрана строки подсказки , в которой содержется информация о функциональных клавишах. Пользователь может вводить и редактировать текст , задавать формат текста (левый отступ , число символов в строке) , центровать нужную строку , осуществлять поиск нужной фразы по шаблону , считать необходимый файл с диска , записать текст в файл на диске , выйти из прграммы.

**9 Тестирование программы**

Для полной проверки работоспособности программы необходимо её протестировать.

План тестирования программы "Текстовый редактор" состоит в следующем:

1) Создание отдельно от программы файла с исходным текстом.

2) Загрузка программы tr.exe.

3) Загрузка файла test.in.

4) Редактирование текста.

5) Изменение параметров строки и центрирование отдельных строк текста.

6) Создание шаблона для поиска и поиск фразы в тексте по этому шаблону.

7) Запись отредактированного текста в файл test.out.

8) Выход из программы.

Исходный тестовый файл test.in содержит в себе следующий текст:

Файл для тестирования программы tr.exe

эта программа предназначена для ввода и редактирования текста

**Тестирование программы**

Далее в пункте "Результаты работы программы" приводится распечатка тестирования программы по пунктам плана тестирования и результатов тестирования.

10 Результаты работы программы

1) Создание отдельно от программы файла с исходным текстом:

Файл test.in содержит в себе следующий текст:

Файл для тестирования программы tr.exe

эта программа предназначена для ввода и редактирования текста

ТЕСТИРОВАНИЕ ПРОГРАММЫ

2) Загрузка программы tr.exe:

ESC - Меню, F4 - Поиск, Alt-C - центровка 0 0

3) Загрузка файла test.in:

1) Сохранить файл

2) Считать файл

3) Ввод шаблона

4) Параметры

5) Выход из программы

Введите имя файла: test.in а 0 0

4) Редактирование текста:

Файл для тестирования программы tr.exe

Эта программа предназначена для ввода и редактирования текста

ТЕСТИРОВАНИЕ ПРОГРАММЫ

ESC - Меню, F4 - Поиск, Alt-C - центровка 0 0

5) Изменение параметров строки и центрирование отдельных строк текста:

Файл для тестирования программы tr.exe

эта программа предназначена для ввода и редактирования текста

ТЕСТИРОВАНИЕ ПРОГРАММЫ

1) Сохранить файл

2) Считать файл

3) Ввод шаблона

4) Параметры

5) Выход из программы

Введите левый отступ: иск, A5 а 0 0

Файл для тестирования программы tr.exe

эта программа предназначена для ввода и редактирования текста

ТЕСТИРОВАНИЕ ПРОГРАММЫ

1) Сохранить файл

2) Считать файл

3) Ввод шаблона

4) Параметры

5) Выход из программы

Введите количество символов 40 а 0 0

Центровка 3 строки:

Файл для тестирования программы tr.exe

эта программа предназначена для ввода и редактирования текста

ТЕСТИРОВАНИЕ ПРОГРАММЫ

ESC - Меню, F4 - Поиск, Alt-C - центровка 0 3

6) Создание шаблона для поиска и поиск фразы в тексте по этому шаблону:

Файл для тестирования программы tr.exe

эта программа предназначена для ввода и редактирования текста

ТЕСТИРОВАНИЕ ПРОГРАММЫ

1) Сохранить файл

2) Считать файл

3) Ввод шаблона

4) Параметры

5) Выход из программы

Введите шаблон для поиска: текст

эта программа предназначена для ввода и редактирования текста

ТЕСТИРОВАНИЕ ПРОГРАММЫ

ESC - Меню, F4 - Поиск, Alt-C - центровка 56 0

7) Запись отредактированного текста в файл test.out:

Файл для тестирования программы tr.exe

эта программа предназначена для ввода и редактирования текста

ТЕСТИРОВАНИЕ ПРОГРАММЫ

1) Сохранить файл

2) Считать файл

3) Ввод шаблона

4) Параметры

5) Выход из программы

Введите имя файла: test.out а 0 0

8) Выход из программы:

Файл для тестирования программы tr.exe

эта программа предназначена для ввода и редактирования текста

ТЕСТИРОВАНИЕ ПРОГРАММЫ

1) Сохранить файл

2) Считать файл

3) Ввод шаблона

4) Параметры

5) Выход из программы

ESC - Меню, F4 - Поиск, Alt-C - центровка 0 0

**Список литературы**

1.Бьярн Страуструп. Язык программирования С++.в двух частях. Пер. с англ. Киев:"ДиаСофт",1993.-296 с.,ил.

2.Корриган Джон Компьютерная графика: Секреты и решения: Пер с англ. -М.:Энтроп, 1995. - 352 с., ил.

ПРИЛОЖЕНИЕ 1

Распечатка программы

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include <string.h>

#include <iostream.h>

// Указатель на начало текста

char \* text;

// Длинна текста

long len;

// Вывод на экран

// Входные данные:

// Адрес начала верхней строки

// Позиция курсора на ~бесконечном~ по ширине экране (x,y)

void output(char \* a,int x,int ky) {

// Количество непечатаемых символов

int np=(x/80)\*80;

// Реальная координата курсора на экране

int rx=x-np;

// Координата Y вывода

int y;

char \* ot=a;

clrscr();

int n;

y=0;

do {

// Пропуск непечатаемых символов

n=np;

while ((n>0)&&(\*ot!=13)) {

ot++;

n--;

}

// Если есть, что печатать на этой строке - печать

if (\*ot!=13) {

// Печать строки

n=80;

while ((n>0)&&(\*ot!=13)) {

cout<<\*ot;

ot++;

n--;

}

// Если строка окнчилась раньше края экрана - перевести строку

if (\*ot==13) {

cout<<(char) 13<<(char) 10;

ot++;

if (\*ot==10) ot++;

} else {

while (\*ot!=13) ot++;

if (\*ot==13) ot++;

if (\*ot==10) ot++;

}

y++;

} else {

// Если печатать нечего - перевод строки

cout<<(char) 13<<(char) 10;

ot++;

y++;

if (\*ot==10) ot++;

}

// Пока не распечатаны все строки

} while (y<23);

// Вывод подсказки и курсора

gotoxy(5,25);

textcolor(14);

cprintf(

"ESC - Меню, F4 - Поиск, Alt-C - центровка %4d %4d ",x,ky);

textcolor(15);

gotoxy(rx+1,ky+1);

}

// Ввод строки

// s - указатель на строку

// lng - Максимальная длинна

// x,y - координаты

void SInput(char \* s,int lng,int x,int y) {

lng--;

char rs[100]; // Временная строка

strcpy(rs,s); // Копирование резерва строки

int n; // Счетчик

char c; // Символ

// Вывод пустой строки

textbackground(2);

textcolor(15);

gotoxy(x,y);

for (n=0;n<lng;n++) cprintf(" ");

n=strlen(s); // Положение курсора

do {

// Вывод строки

gotoxy(x,y);

cprintf("%s ",s);

gotoxy(x+n,y);

c=getch(); // Считывание символа с клавиатуры

if ((c!=13)&&(c!=27)&&(c!=8)) {

// Нажата символьная клавиша

if (n<(lng-1)) {

// Добавление символа

s[n+1]=0;

s[n]=c;

n++;

}

} else {

if (c==8) {

// Нажата клавиша BackSpace

if (n>0) {

n--;

s[n]=0;

}

}

}

// Выход, если ESC или ENTER

} while ((c!=27)&&(c!=13));

if(c==27) strcpy(s,rs); // Восстановление строки

// Стирание строки

textbackground(0);

textcolor(7);

gotoxy(x,y);

for (n=0;n<lng+1;n++) cprintf(" ");

}

// Чтение файла

void load(char \* p) {

char fname[25];

fname[0]=0;

FILE \* f;

gotoxy(1,25);

printf(" Введите имя файла: ");

SInput(fname,20,21,25);

f=fopen(fname,"rb");

len=fread(p,32767,1,f);

fclose(f);

}

// Запись в файл

void save(char \* p) {

char fname[25];

fname[0]=0;

char \* pr;

pr=p;

len=0;

while((len<32767)&&((\*pr!=13)||(\*(pr+1)!=13))) {

len++;

pr++;

}

FILE \* f;

gotoxy(1,25);

printf(" Введите имя файла: ");

SInput(fname,20,21,25);

f=fopen(fname,"wb");

fwrite(p,len,1,f);

fclose(f);

}

// Уничтожение информации - заполнение кодом "Перевод строки"

void reset(char \* a,int n) {

for (int m=0;m<n;m++) \*(a+m)=13;

}

// Вверх на строку

char \* up(char \* a,char \* b) {

if (a==b) return b;

b--;

while ((a!=b)&&(\*b!=13)) b--;

if (a==b) return b;

if (\*b==13) b--;

if (a==b) return b;

while ((a!=b)&&(\*b!=13)) b--;

if (\*b==13) b++;

if (\*b==10) b++;

return b;

}

// Вниз на строку

char \* down(char \* a,char \* b) {

while (((a+32767)>=b)&&(\*b!=13)) b++;

if (\*b==13) b++;

if (\*b==10) b++;

return b;

}

// Обработка меню

int menu(int x,int y,char \* capt) {

int n,m; // Счетчики

int num; // Количество пунктов

int k; // Выбранный пункт

char \* pt; // Временный указатель на символ

char c; // Считанный с клавиатуры символ

// Вычисляем количество пунктов

num=strlen(capt)/20;

// Курсор на нулевой элемент

k=0;

// Бесконечный цикл обработки

for (;;) {

// Вывод меню

pt=capt;

for (n=0;n<num;n++) {

gotoxy(x,y+n);

// Закраска пункта, на который указывает курсор

if (n==k) {

// Закраска

textbackground(12);

textcolor(14);

} else {

// Нормальный

textbackground(3);

textcolor(1);

}

cprintf("%d) ",n+1);

for (m=0;m<20;m++) cprintf("%c",\*(pt++));

}

textbackground(3);

textcolor(1);

// Опрос клавиатуры

c=getch();

if (!c) c=getch();

// Проверка, не нажата ли клавиша с цифрой

if (((c-'1')>=0)&&((c-'1')<num)) {

// Установка указателя в зависимости от нажатой цифры

k=c-'1';

// Запись в буфер клавиатуры символа ENTER

ungetch(13);

} else {

// Анализ

switch(c) {

// Вверх

case (72):

if (k>0) k--; else k=num-1;

break;

// Вниз

case (80):

if (k<(num-1)) k++; else k=0;

break;

// Выход по ESC - возвращается -1

case (27):

return -1;

// Выход по ENTER - возвращается номер пункта

case (13): return k;

}

}

}

}

// Добавление символа в текст

void add(char \* st,char \* a,int x,int y,char c) {

int n;

int d;

n=y;

char \* p;

// Перемещение указателя к началу строки

while (n>0) {

while ((a<(st+32767))&&(\*a!=13)) a++;

if (\*a==13) a++;

if (\*a==10) a++;

n--;

}

// Подсчет символов в строке

n=0;

while ((\*(a+n)!=13)&&((a+n)<(st+32767))) n++;

if (n<x) {

// Если производится вставка в строку

a=a+n;

d=x-n;

p=st+32767;

while ((p-d)!=a) {

\*p=\*(p-d-1);

p--;

}

p=a;

while (d!=0) {

\*p=' ';

d--;

p++;

}

\*p=c;

} else {

// Если производится добавление к строке не только символа,

// но и пробелов до него

a=a+x;

p=st+32767;

while (p!=a) {

\*p=\*(p-1);

p--;

}

\*a=c;

}

}

// Поиск конца строки - фактически возвращает длину строки

int end(char \* st,char \* a,int,int y) {

int n;

n=y;

char \* p;

while (n>0) {

while ((a<(st+32767))&&(\*a!=13)) a++;

if (\*a==13) a++;

if (\*a==10) a++;

n--;

}

n=0;

while ((\*(a+n)!=13)&&((a+n)<(st+32767))) n++;

return n;

}

// Удаление символа

void del(char \*st,char \* a,int x,int y) {

int n;

int d;

n=y;

char \* p;

// Поиск необходимого места в тексте

while (n>0) {

while ((a<(st+32767))&&(\*a!=13)) a++;

if (\*a==13) a++;

if (\*a==10) a++;

n--;

}

n=0;

while ((\*(a+n)!=13)&&((a+n)<(st+32767))) n++;

if (n<x) x=n;

a+=x;

p=a;

n=\*p;

// Удаление символа

while (a!=(st+32767)) {

\*a=\*(a+1);

a++;

}

// Если был удален "Перевод строки" и остался

// "Возврат каретки" либо наоборот, то удаляется еще один символ

if ((\*p==13)&&(n==10)||(\*p==10)&&(n==13)) {

while (p!=(st+32767)) {

\*p=\*(p+1);

p++;

}

}

}

// Поиск фрагмента в тексте, начиная с текущего положения курсора

void find(char \* st,char \* \* scrs,int \* x,int \* y,char \* b) {

char \* a=\*scrs;

int n;

int k;

int d;

n=\*y;

char \* p;

// Поиск текущего места

while (n>0) {

while ((a<(st+32767))&&(\*a!=13)) a++;

if (\*a==13) a++;

if (\*a==10) a++;

n--;

}

n=0;

while ((\*(a+n)!=13)&&((a+n)<(st+32767))) n++;

if (n<\*x) \*x=n;

a+=\*x;

// Поиск фрагмента

do {

a++;

k=1;

for (n=0;n<strlen(b);n++)

if (a[n]!=b[n]) k=0;

} while ((k!=1)&&(a<(st+32767)));

if (k==1) {

// Если фрагмент найден - перемещение курсора

\*x=0;

\*y=0;

while ((\*a!=13)&&(a!=st)) {

a--;

(\*x)++;

}

if (\*a==13) {

a++;

(\*x)--;

}

if (\*a==10) {

a++;

(\*x)--;

}

\*scrs=a;

}

}

// Ввод числа

// x,y - координаты

long DInput(int x,int y) {

char s[100]; // Временная строка

char \*stpe;

long rz;

s[0]=0;

int n; // Счетчик

char c; // Символ

// Вывод пустой строки

textbackground(2);

textcolor(15);

gotoxy(x,y);

for (n=0;n<15;n++) cprintf(" ");

n=strlen(s); // Положение курсора

do {

// Вывод строки

gotoxy(x,y);

cprintf("%s ",s);

gotoxy(x+n,y);

c=getch(); // Считывание символа с клавиатуры

if ((c!=13)&&(c!=27)&&(c!=8)) {

// Нажата символьная клавиша

if (n<14) {

// Добавление цифры

if ((c>='0')&&(c<='9')) {

s[n+1]=0;

s[n]=c;

n++;

}

}

} else {

if (c==8) {

// Нажата клавиша BackSpace

if (n>0) {

n--;

s[n]=0;

}

}

}

// Выход, если ESC или ENTER

} while ((c!=27)&&(c!=13));

if(c==27) return 0;

// Стирание строки

textbackground(0);

textcolor(7);

gotoxy(x,y);

for (n=0;n<15;n++) cprintf(" ");

sscanf(s,"%ld",&rz);

return rz;

}

// Центровка текущей строки

void centr(char \*st,char \* a,int y,int left,int right) {

int n;

int d;

n=y;

// Поиск текущего местоположения

while (n>0) {

while ((a<(st+32767))&&(\*a!=13)) a++;

if (\*a==13) a++;

if (\*a==10) a++;

n--;

}

// Удаление пробелов в начале строки

while ((\*a==32)&&(!((\*a==13)||(a==(st+32767))))) del(st,a,0,0);

// Подсчет длины строки

n=0;

while ((\*(a+n)!=13)&&((a+n)<(st+32767))) n++;

// Определение количества добавляемых пробелов

if ((n>(right-left))||(n==0)) return;

d=left+((right-left)-n)/2;

// Добавление пробелов

for (n=0;n<d;n++) add(st,a,0,0,' ');

}

// Основная программа

void main() {

// Границы текста

int left=0;

int right=60;

// Маска для поиска

char masks[40];

masks[0]=0;

// Положение курсора

char \* scrs;

int x,y;

char c;

int n;

// Выделение памяти под текст

text=new char[32767];

len=0;

reset(text,32767);

scrs=text;

x=0;y=0;

// Цикл обработки данных

for (;;) {

textbackground(1);

textcolor(15);

// Вывод данных на экран

output(scrs,x,y);

// Считывание символа

c=getch();

// Обработка

switch (c) {

case (27):

// Вывод меню

switch (menu(30,5," Сохранить файл "

" Считать файл "

" Ввод шаблона "

" Параметры "

" Выход из программы "

"\x0")) {

case (0):

save(text);

break;

case (1):

reset(text,32767);

load(text);

break;

case (2):

gotoxy(1,25);

printf(" Введите шаблон для поиска: ");

SInput(masks,40,30,25);

break;

case (3):

gotoxy(1,25);

printf(" Введите левый отступ: ");

left=DInput(30,25);

gotoxy(1,25);

printf(" Введите количество символов в строке: ");

right=DInput(30,25);

break;

case (4):

textbackground(0);

textcolor(7);

clrscr();

return;

}

break;

case (0):

// Расширенный код - считывание значащей части

c=getch();

// обработка

switch(c) {

// Alt-C

case (46):

centr(text,scrs,y,left,right);

break;

// F4

case (62):

find(text,&scrs,&x,&y,masks);

break;

// Home

case (71):

x=0;

break;

// End

case (79):

x=end(text,scrs,x,y);

break;

// Delete

case (83):

del(text,scrs,x,y);

break;

// Стрелки перемещения курсора

case (72):

if (y>0) y--;

else scrs=up(text,scrs);

break;

case (80):

if (y<22) y++;

else scrs=down(text,scrs);

break;

case (77):

if (x<32767) x++;

break;

case (75):

if (x>0) x--;

break;

}

break;

// Enter

case (13):

add(text,scrs,x,y,10);

add(text,scrs,x,y,13);

if (y<22) y++; else down(text,scrs);

x=0;

break;

// BackSpace

case (8):

if (x>0) {

del(text,scrs,x-1,y);

x--;

}

break;

// Tab

case (9):

for (n=0;n<(5-(x-x/5));n++) add(text,scrs,x,y,' ');

x=x+(5-(x-x/5));

break;

// Остальные символы

default:

add(text,scrs,x,y,c);

x++;

}

}

}

ПРИЛОЖЕНИЕ 2

Алгоритм работы программы