**Файлы данных и функции ввода – вывода.**

Под файлом далее будем понимать поименованную часть памяти. В функциях, работающих с файлами, язык Си “рассматривает” файл как структуру. Эта структура описана в файле stdio.h. Наименование шаблона файла – FILE.

Поток – это файл данных или физическое устройство (принтер, дисплей и т.п.), с которым программист работает с помощью указателя на объект типа FILE. Так, функция, работающая с потоком данных, возвращает в качестве значения, значение указателя на объект типа FILE.

Открытие файла**: fopen\_s()**

Фрагмент программы:

FILE \*in;

fopen\_s(&in,“test.dat”,”r”);

Функцией fopen() управляют три основных параметра:

1. Имя файла, который следует открыть (test.dat).
2. Описание того, как должен использоваться файл
   * “r” – чтение существующего файла;
   * “w’ – запись;
   * “a” – добавление в конец файла (если файла нет, он создается);
   * “r+” – чтение и запись в файл (файл должен существовать);
   * “w+” – чтение и запись в файл (если файл уже существует, то его содержимое уничтожается);
   * “a+” – чтение и добавление в файл.

Кроме этого различают еще текстовый (t) и двоичный (b) потоки. Если при открытии файла не указан тип потока, то по умолчанию файл обрабатывается системой как текстовый файл.

* + 1. Указатель на файл (in). Его значение возвращается функцией и с этого момента программа ссылается на файл при помощи указателя, а не по имени файла.

Закрытие файла: **fclose()**

**Пример:**

**fclose(in);**

Аргументом функции является указатель на файл, а не имя файла. Функция возвращает 0, если файл закрыт успешно, и -1 в противном случае.

Ввод и вывод символа из файла: **getc(), putc()**

**Пример:**

ch=getc(in); /\* ввод символа \*/

putc(ch,out); /\* вывод символа \*/

**Пример:/\* Иллюстрация работы функций fopen(), fclose() \*/**

**/\* Чтение из файла. Имя файла записывается в тело программы \*/**

#include <stdio.h>

#include<windows.h>

#include <stdlib.h>

#include <string.h>

int main()

{

char cStr1[] = "Необходимо имя файла \n";

char cStr2[] = "Невозможно открыть файл";

FILE \*in, \*out;

int ch;

char name[20];

int count = 0;

CharToOem(cStr1, cStr1);

CharToOem(cStr2, cStr2);

if (fopen\_s(&in, "c:\\Test.txt", "r") == 0)

{

strcpy\_s(name, 20, "c:\\Programm\\Out");

printf("%s\n", name);

strcat\_s(name, 20, ".txt");

printf("%s\n", name);

fopen\_s(&out, name, "w+");

while ((ch = fgetc(in)) != EOF)

if (count++ % 3 == 0)

putc(ch, out);

fclose(in);

fclose(out);

}

else

printf("Error!!!\n");

system("pause");

return 0;

}

**/\* Чтение из файла. Имя файла записывается в командной строке \*/**

#include <stdio.h>

#include <string.h>

#include <windows.h>

#include<stdlib.h>

int main(int argc, char \*argv[])

{

char cStr1[] = "Необходимо имя файла \n";

char cStr2[] = "Невозможно открыть файл";

FILE \*in, \*out;

int ch;

char name[20];

int count = 0;

CharToOem(cStr1, cStr1);

CharToOem(cStr2, cStr2);

if (argc < 2)

printf("%sn", cStr1);

else

{

if ((fopen\_s(&in, argv[1], "r")) == 0)

{

strcpy\_s(name, 20, "c:\\Programm\\res.txt");

fopen\_s(&out, name, "w+");

while ((ch = getc(in)) != EOF)

if (count++ % 3 == 0)

{

putc(ch, out);

}

fclose(in);

fclose(out);

}

else

printf("%s %s\n", cStr2, argv[1]);

}

system("pause");

return 0;

}

**Замечание:** Необходимо иметь текстовый файл, имя которого задается при запуске программы на выполнение. Содержимое файла в примере: “Даже Эдди нас опередил с детским хором”.В результате выполнения программы появляется новый файл с текстом …..

Функции ввода и вывода данных из файла: **fscanf\_s(), fprintf().** Эти две функции работают также как и функции printf(), scanf(), но не со стандартными устройствами ввода и вывода, а с файлами. При работе с этими функциями необходим еще один дополнительный аргумент – указатель на файл.

FILE \*in, \*out;

if ((fopen\_s(&in,"c:\\Programm\\dat.txt", "r")) == 0)

{

fscanf\_s(in, "%d", &n);

…

fprintf(out, "%6.2f ", x[i]);

**Работа с динамическими массивами**

В Си реализовано статитическое выделение памяти: объем необходимой памяти известен на этапе компиляции.

Можно организовать динамическое выделение памяти: выделение памяти во время выполнения программы.

Прототипы функций, управляющих выделением памяти, находятся в файле ***stdlib.h***

1. Функции выделения памяти.
2. Функции изменения объема выджеляемой памяти.
3. Функции освобождения памяти.

***Функции выделения памяти:***

1. calloc() – динамически выделяет блок памяти и инициализирует его нулями.
2. malloc() - динамически выделяет блок памяти без инициализации его содержимого.

Формат функций:

1. calloc(size\_t n, size\_t s), где n – количество элементов, под хранениекоторых выделяется память; s - размер пямяти в байтах для хранения одного элемента.
2. malloc(size\_t s), где s – размер выделяемой памяти в байтах.

Пример:

int \*p;

p = calloc(10, sizeof(int));

int \*p;

p = malloc(10\* sizeof(int));

*Выделение памяти с инициализацией :*

int \* p=(int\*) calloc(10, sizeof());

***Функция перераспределения памяти:***

realloc(void \*p, size\_t s) , где p – указатель на блоквыделенной памяти, s – новый размер блока выделяемой памяти.

***Функция освобождения памяти:***

free(void \* P);

***Выделение памяти по массив***

***Пример:*** /\* Выделение памяти и ввод данных в двумерный массив x(m, n) \* /

int \*\* x;

x= (int \*\*) calloc(n, sizeof(int))

{

for(i=0; i<n; i++)

x[i]=(int \*) calloc(m, sizeof(int));

for(j=0; j<m; j++)

{

scanf\_s(“%d”,&x[i][j]);

}

}

Задание: Написать программу:

**/\* Нахождение выборочного мат. ожидания, стандартного отклонения, коэффициента корреляции, значений однофакторной линейной регрессии. Данные хранятся в файле dat.txt, результаты выполнения программы записываются в файл dat1.txt \*/**

****

***Интерполирование по методу наименьших квадратов***

Пусть мы ищем приближающий полином первой степени: 

Рассмотрим функционал, который надо минимизировать: , где .

Таким образом, надо найти два коэффициента  так, чтобы минимизировать функционал .

Для этого найдем частные производные по параметрам и приравняем их к нулю:



,

где .

Таким образом, имеем два уравнения и два неизвестных, и неизвестные входят в уравнения линейно. То есть, имеем СЛАУ (систему линейных алгебраических уравнений) 2-го порядка.

Запишем в матричном виде:

 где 

Домножая слева левую и правую части на , получим решение , то есть вектор коэффициентов, минимизирующий рассматриваемый функционал в смысле метода наименьших квадратов: 

Пусть x – рост, y – вес, представленные выборочными значениями. Пусть в файле dat.txt хранятся выборки объемом n=10 для роста и веса соответственно:

155 170 175 182 177 183 169 180 177 168

59 74 80 87 83 90 75 94 85 77 .

Тогда реультаты выполнения программы (файл dat1.txt):

M1=173.60 M2= 80.40 Sr1= 7.98 Sr2= 9.43 R= 0.96

x:155.00 170.00 175.00 182.00 177.00 183.00 169.00 180.00 177.00 168.00

y: 59.00 74.00 80.00 87.00 83.00 90.00 75.00 94.00 85.00 77.00

yr: 59.31 76.32 81.99 89.92 84.26 91.06 75.18 87.66 84.26 74.05