***Функции ввода / вывода (дополнение)***

С функциями ввода/вывода ассоциирован заголовок <stdio.h>. Этот заголовок определяет некоторые макросы и типы, которые используются файловой системой. Наиболее важным из них является тип FILE, который используется для объявления указателя на файл. Два других часто используемых типа — size\_t и fpos\_t. Тип size\_t, представляющий собой некоторую разновидность целых без знака, — это тип результата, возвращаемого функцией sizeof. Тип fpos\_t определяет объект, который однозначно задает каждую позицию в файле. Самым популярным макросом, определенным в этом заголовке, является макрос EOF, значение которого указывает на конец файла. Другие типы данных и макросы, определенные в заголовке <stdio.h>, описаны вместе с функциями, с которыми они связаны.

Многие функции ввода/вывода при возникновении ошибки присваивают встроенной глобальной переменной целого типа errno определенное значение. Анализ этой переменной поможет программе получить более подробную информацию о возникшей ошибке. Значения, которые может принимать переменная errno, зависят от конкретной реализации компилятора.

**fseek– установка позиции в потоке данных.**

**Синтаксис:**

#include < stdio.h >   
int fseek(FILE \*stream, long int offset, int whence);

**Аргументы:**

stream – указатель на управляющую таблицу потока данных.   
offset – смещение позиции.   
whence – точка отсчета смещения.

**Возвращаемое значение:**

0 – при успешной установки позиции.   
  
Отличное от нуля значение, если при работе функции произошли ошибки. При этом переменной ernno будет присвоен код ошибки:  
         [EINVAL] – неверное значение аргумента whence  
         [ESPIPE] – недопустимое значение аргумента offset

**Описание:**

Функция fseek() устанавливает позицию в потоке данных, заданным аргументом stream. Относительно установленной позиции будет осуществляться чтение и запись данных.   
  
Точка отсчета устанавливаемой позиции определяется аргументом whence, который может принимать значения:  
         SEEK\_SET – смещение отсчитывается от начала файла  
         SEEK\_CUR – смещение отсчитывается от текущей позиции  
         SEEK\_END – смещение отсчитывается от конца файла

|  |  |
| --- | --- |
| **Код** | **Положение в файле** |
| 0 | начало файла |
| 1 | текущая позиция |
| 2 | конец файла |

Смещение задается аргументом offset, причем положительное значение аргумента означает смещения вправо от указанной аргументом whence позиции, отрицательное – смещение влево.   
  
Для двоичных потоков данных, смещение (offset) – это количество байт.   
  
Для текстовых потоков данных смещение (offset) должно быть равно 0, либо получено с помощью функции ftell(), при этом точка отсчета (whence) должна иметь значение SEEK\_SET.

/\* Чтение и запись в файл с функцией feof()\*/

#include <stdio.h >

#include <windows.h>

#include <stdlib.h>

#include<locale.h>

int main()

{

FILE \*in, \*out;

long num = 10L;

/\* #include<locale.h> \*/

setlocale(LC\_ALL, "rus");

// setlocale(0, "");

/\*#include <windows.h>\*/

// SetConsoleOutputCP(1251);

printf("Читаем на русском языке!\n");

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

printf("Не открыть файл \n");

else

{

fopen\_s(&out,"c:\\Programm\\data1.txt", "w");

fseek(in, num, 0);

while (!feof(in))

{

fprintf(out, "%c", getc(in));

}

fclose(in);

fclose(out);

}

system("pause");

return 0;

}

***Метод Монте-Карло***

/\* Приближенное вычисление числа пи \*/

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

#include<locale.h>

#define P 3.14159265

int main()

{

struct coord {

double x;

double y;

} z;

long i, j, n, n1 = 1e+8;

double S, Sr = 0;

srand((unsigned)time(NULL));

setlocale(LC\_ALL, "rus");

printf("Введите число циклов моделирования : \n");

scanf\_s("%d", &n);

time\_t beg\_t, end\_t;

beg\_t = time(NULL);

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

S = 0;

for (i = 0; i < n1; i++)

{

z.x = (double)rand() / (double)RAND\_MAX;

z.y = (double)rand() / (double)RAND\_MAX;

z.x -= 0.5;

z.y -= 0.5;

if ((z.x\*z.x + z.y\*z.y) <= 0.25)

S++;

}

S /= (double)n1;

printf("S=%10.8f\n", S);

Sr += S;

}

end\_t = time(NULL);

printf("T=%f sec\n", difftime(end\_t, beg\_t));

printf("Int=%10.8f\tPi=%10.8f\n", Sr / (double)n, 4 \* Sr / (double)n);

system("pause");

return 0;

}