acm.mipt.ru

олимпиады по программированию на Физтехе

Раздел «Язык Си» . ReadAndWrite :

Чтение и печать целого числа без использования stdio

Используя функции scanf и printf, можно считывать форматированный ввод и печатать форматированный выход. Их возможности очень велики.

scanf и printf не есть команды языка C. Они являются функциями, определенными в стандартной библиотеке ANSI C.

Возникает вопрос: Как они описаны в этой библиотеке?

Ответ: Есть более низкоуровневые функции для считывания и печати данных, но они платформозависимые (зависят от архитектуры комптютера и операционной системы).

В OS Linux присутствуют системные вызовы read и write, которые читают (пишут) указанное количество байт из (в) указанный поток.

Аргументы у этих системных вызовов одинаковые, их три штуки:

- идентификатор потока ввода/вывода (дескриптор потока)
- адресс, куда нужно помещать или откуда нужно брать байты
- максимальное количество байт, которое мы хотим считать (напечатать)

Стандартный поток ввода имеет идентификатор (дескриптор потока) равный 0. Стандартный поток вывода имеет идентификатор 1.

Возвращаемое значение равно количеству успешно записанных (считанных) байт.

```
#include <sys/syscall.h>
#define N 100

char buffer[N];

void print_int (int);
int  scan_int  (void);


main()
{
        int a = scan_int();
        int b = scan_int();

        print_int(a+b);
        return 0;
}

void print_int(int a)
{
        char minus = '-';
        if(a < 0) {
                write(1, &minus,  1);
                a = -a;
        }
        int i = N;
        buffer[--i] = '\n';
        while(a)
        {
                buffer[--i] = '0' +  (a % 10);
                a /= 10;
        }
        write(1, &(buffer[i]), N - i);

}
```

Поиск
Поиск
Раздел «Язык Си»
Главная
Зачем учить С?
Определения
Инструменты:
Поиск
Изменения
Index
Статистика
Разделы
Информация
Алгоритмы
Язык Си
Язык Ruby
Язык Ассемблера
El Judge
Парадигмы
Образование
Сети
Objective C
Logon>>

```
int scan_int()
{
        int i = 0, res = 0, sign = 1;
        int n = read(2, buffer, N);
        if (buffer[i] == '-') {
            sign = -1;
            i++;
        }
        for(; i < n ; i++)
        {
                if( buffer[i] >= '0' && buffer[i] < '9' )
                {
                        res *= 10;
                        res += buffer[i] - '0';
                } else {
                        break;
                }
        }
        return res;

}
```

Программа сложения двух целых чисел

```
section .bss
    buffer resb 20

section .text
global _start
_start:
    call scan_int
    push eax

    call scan_int
    push eax

    call print_int

    mov eax,1
    mov ebx, 0
    int 0x80

scan_int:

    push ebp
    mov ebp,esp
    xor eax,eax
    xor edi, edi
    XOR ESI,ESI

.lop:
    mov esi,eax

    mov eax,3
    mov ebx,2
    mov ecx, buffer
    mov edx,1
    int 0x80

    mov eax,esi

    mov edx, [ecx]

    cmp dl, 0xA
    je .end

    mov ecx,edx
    mov esi, 10
```

```
    mul esi
    mov edx,ecx

    sub edx, '0'

    add eax, edx
    inc edi
    jne .lop

.end:
    leave
    ret


print_int:

    push ebp
    mov ebp,esp
    mov eax,[esp+12]
    mov ecx,[esp+8]
    add ecx,eax
    xor edx,edx
    mov esi,10
    mov edi,18

    mov byte [buffer+18],0xA
    mov byte [buffer+19],0

.loop
    mov eax,ecx
    xor edx,edx
    div esi
    mov ecx,eax

    add edx,'0'
    dec edi
    mov byte [buffer+edi],dl
    cmp ecx,0
    jne .loop

    mov eax,4
    mov ebx,1
    mov ecx,buffer
    add ecx,edi
    mov edx,19
    sub edx,edi
    int 0x80

    leave
    ret
```