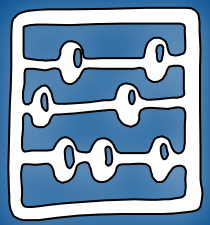
Алгоритмы и Алгоритмические Языки

Семинар #2:

1. Результаты работы по системам счисления;
2. Программные контракты и проверка ввода;
3. Представление целых чисел в памяти;
4. Контест#0: тестовый контест.

12.09.2023

Результаты работы по системам счисления



Работа над ошибками

1.3. $0.(3)_{13} = ???_{10}$

3.2. $0.91_{10} = ???_{7}$

HSE University

Программные контракты и проверка ввода

Условие задачи в системе

Problem 00-2: A+B

На стандартном потоке ввода задаются два целых числа, не меньшие -32000 и не большие 32000.

На стандартный поток вывода напечатайте сумму этих чисел.

Числа задаются по одному в строке. Пробельные символы перед числом и после него отсутствуют. Пустые строки в вводе отсутствуют.

Examples

Input

```
1
2
```

Output

```
3
```

Простое решение задачи

```c
#include <stdio.h>

int main(void)
{
    // Declare variables:
    short num1, num2 = 0;

    // Parse program input:
    scanf("%hd\n%hd", &num1, &num2);

    // Print result:
    printf("%hd\n", num1 + num2);

    return 0;
}

```

Запуск простого решения

```
❯ cd ~/path/to/repository/examples/02_program_contracts
❯ gcc adder.c -o adder // Компиляция кода
❯ ./adder
1 // Жёлтым обозначен ввод данных от пользователя
2
3 // Зелёным – вывод программы
❯ ./adder
error
22940
```

При нарушении контракта программа выводит мусор!
Как пользователю понять, какой контракт был нарушен?

Проверка вводимых данных

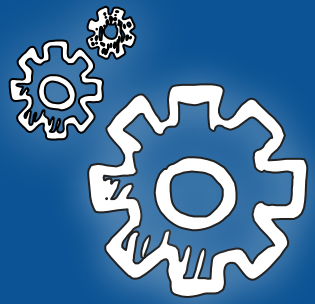
Обратимся к документации на **scanf**.
Можно получить её как и с cppreference, так и из консоли:

```
❯ man scanf
SCANF(3)                Linux Programmer's Manual
...
RETURN VALUE
    On success, these functions return the  number  of  input
    items  successfully matched  and  assigned;  this can be
    fewer than provided for, or even zero, in the event of an
    early matching failure.
```

Проверка вводимых данных

```c
#include <stdio.h>

int main(void)
{
    short num1, num2 = 0;

    // Parse program input:
    int num_inputs = scanf("%hd\n%hd", &num1, &num2);
    if (num_inputs < 2)
    {
        printf("adder: expected input \"<addentum1>\\n<addentum2>\\n\"\n");
        return 1;
    }

    // Print result:
    printf("%hd\n", num1 + num2);

    return 0;
}
```

Представление целых чисел в памяти компьютера



Выход за границы представимости

Каково поведение программы при работе с большими числами?

```
❯ cd ~/path/to/repository/examples/02_program_contracts
❯ gcc adder.c -o adder
❯ ./adder
1
2
3
❯ ./adder
32500
32500
-536
```

Представление целых чисел

Тип данных	Типичный размер	Представимые значения
unsigned short	16 бит	от 0 до 65535
short		от -32768 до 32767
unsigned int	32 бита	от 0 до $2^{32}-1$
int		от -2^{31} до $2^{31}-1$
unsigned long	64 бита	от 0 до $2^{64}-1$
long		от -2^{63} до $2^{63}-1$

Согласно стандарту, если в результате операции получается непредставимое значение, то фактический результат – зависит от представления чисел и неопределён (Undefined Behavior).

Инверсный дополнительный код

Кодирование отрицательных чисел “знак+модуль”:

$value = (-1)^{sign} \times |value|$

$+1_{10} \leftrightarrow 00000001$ $\quad -0_{10} \leftrightarrow 10000000$ $\quad -127_{10} \leftrightarrow 11111111$

$+0_{10} \leftrightarrow 00000000$ $\quad -1_{10} \leftrightarrow 10000001$ $\quad +127_{10} \leftrightarrow 01111111$

Проблемы такого представления?

Инверсный дополнительный код:

Старший бит – знаковый.

$-x \leftrightarrow \sim x + 1$

$+0_{10} \leftrightarrow 00000000$ $\quad -0_{10} \leftrightarrow \sim 00000000 + 1 = 11111111 + 1 = 00000000$

$+9_{10} \leftrightarrow 00001001$ $\quad -9_{10} \leftrightarrow \sim 00001001 + 1 = 11110110 + 1 = 11110111$

Выход за границы представимости

```
32500 + 32500 = 0111_1110_1111_0100
              + 0111_1110_1111_0100
              = 1111_1101_1110_1000 = -536
```

Проверка на переполнение:

```c
// Cast numbers to long and check if they overflow on addition:
int num1_ext = num1;
int num2_ext = num2;
int sum_ext = num1_ext + num2_ext;
if (sum_ext < SHRT_MIN || sum_ext > SHRT_MAX)
{
    printf("adder: %hd+%hd is too big to be represented in short", num1, num2);
    return 1;
}
```

Написание тестового контеста

Работа с тестовым контестом

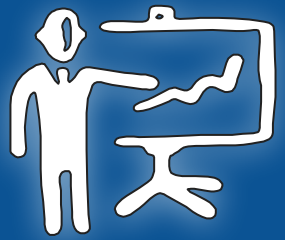
Алгоритм:

1. Переход на сайт с контестами.
2. Получение и **смена пароля**.
3. Прорешивание задачек:
   - **Problem 00-1: Hello world!** – минимальный пример;
   - **Problem 00-2: A+B** – программные контракты;
   - **Problem 00-3: Таблица умножения** – ограничение по времени;
   - **Problem 00-4: Частичный разворот** – расход по памяти.

   Здоровое решение содержит динамическую память.
4. **Контест #1: циклы и битовые операции**

Контест #0 не оценивается!

Вопросы?



Красивые иконки взяты с сайта handdrawngoods.com