

Architektury systemów komputerowych

Lista zadań nr 2

Na zajęcia 12 marca 2020

UWAGA! W trakcie prezentacji rozwiązań należy zdefiniować i wyjaśnić pojęcia, które zostały oznaczone **wytłuszczoną** czcionką. Rozwiązania zadań muszą się trzymać następujących wytycznych:

- Założenia:
 - liczby całkowite są w reprezentacji uzupełnień do dwóch,
 - wartość logiczna prawdy i fałszu odpowiada kolejno wartościom całkowitoliczbowym 1 i 0,
 - przesunięcie w prawo na liczbach ze znakiem jest przesunięciem arytmetycznym,
 - dane typu `int` mają `N` bitów długości,
 - jeśli nie podano inaczej, rozwiązanie musi działać dla dowolnego `N` będącego wielokrotnością 8.
- Zabronione:
 - wyrażenia warunkowe (`?:`) i wszystkie instrukcje (łącznie z `if`) poza przypisaniem,
 - operacja mnożenia, dzielenia i reszty z dzielenia (`*`, `/`, `%`),
 - operacje logiczne (`&&`, `||`, `!`),
 - operatory porównania (`==`, `!=`, `<`, `>`, `<=` i `>=`),
 - rzutowanie – zarówno jawne jak i niejawne,
 - używanie zbędnych nawiasów (zapoznaj się z **priorytetami operatorów w języku C¹**).
- Dozwolone:
 - instrukcja przypisania,
 - operacje bitowe,
 - przesunięcie bitowe w lewo i prawo z argumentem w przedziale `0...N-1`,
 - dodawanie i odejmowanie,
 - stała `N`, stałe własne oraz zdefiniowane w pliku nagłówkowym `limits.h`.

Zadanie 1. Czy poniższe wyrażenia zawsze obliczą się do prawdy dla dwóch dowolnych wartości zmiennych «`x`» i «`y`» typu «`int32_t`»? Jeśli nie to podaj wartości, które prowadzą do obliczenia fałszu.

- `(x > 0) || (x - 1 < 0)`
- `(x & 7) != 7 || (x << 29 < 0)`
- `(x * x) >= 0`
- `x < 0 || -x <= 0`
- `x > 0 || -x >= 0`
- `(x | -x) >> 31 == -1`
- `x + y == (uint32_t)y + (uint32_t)x`
- `x * ~y + (uint32_t)y * (uint32_t)x == -x`

Zadanie 2. Napisz ciąg instrukcji, który bez użycia dodatkowych zmiennych, zamieni miejscami zawartość zmiennych «`x`» i «`y`».

Wskazówka: Spróbuj rozwiązać zadanie samodzielnie, a następnie przeczytaj §2.19 książki „Uczta programistów”.

Zadanie 3. Napisz wyrażenie zawierające wyłącznie zmienne «`x`», «`y`» i «`s`», którego wartością logiczną jest odpowiedź na pytanie czy wykonanie instrukcji «`s = x + y`» spowodowało **nadmiar** (ang. *overflow*) lub **niedomiar** (ang. *underflow*).

Wskazówka: Spróbuj rozwiązać zadanie samodzielnie, a następnie przeczytaj §2.12 książki „Uczta programistów”.

¹https://en.cppreference.com/w/c/language/operator_precedence

Zadanie 4. Zmienne «x» i «y» przechowują liczby typu «uint32_t» składające się z czterech bajtów, tj. $x = \sum_{i=0}^3 x_i \cdot 2^{8i}$ oraz $y = \sum_{i=0}^3 y_i \cdot 2^{8i}$. Jak szybko obliczyć $z = \sum_{i=0}^3 z_i \cdot 2^{8i}$ gdzie $z_i = x_i \oplus y_i$, gdy:

- \oplus jest operacją dodawania,
- \oplus jest operacją odejmowania.

Obliczając wynik należy zapobiec wystąpieniu **przeniesienia** (ang. *carry*) lub **pożyczki** (ang. *borrow*) propagującego się do bardziej znaczącego bajtu.

Wskazówka: Spróbuj rozwiązać zadanie samodzielnie, a następnie przeczytaj §2.17 książki „Uczta programistów”.

Zadanie 5. Uzupełnij ciało funkcji zadeklarowanej następująco:

```
/* Oblicz x * 3 / 4 zaokrąglając w dół. */
int32_t threefourths(int32_t x);
```

Nie można dopuścić do wystąpienia nadmiaru i niedomiaru!

Zadanie 6. Podaj wyrażenie zawierające wyłącznie zmienne «x» i «y», którego wartością logiczną jest wynik porównania «x < y» dla liczb (a) bez znaku (b) ze znakiem.

Wskazówka: Spróbuj rozwiązać zadanie samodzielnie, a następnie przeczytaj §2.11 książki „Uczta programistów”.

Zadanie 7. Podaj fragment kodu, który oblicza funkcję:

$$abs(x) = \begin{cases} x & \text{dla } x \geq 0 \\ -x & \text{dla } x < 0 \end{cases}$$

Skorzystaj z następującej własności: jeśli «b» jest wartością logiczną, to wyrażenie «b ? x : y» można przetłumaczyć do «b * x + !b * y».

Wskazówka: Spróbuj rozwiązać zadanie samodzielnie, a następnie przeczytaj §2.4 książki „Uczta programistów”.

Zadanie 8. Podaj fragment kodu, który oblicza funkcję:

$$sign(x) = \begin{cases} -1 & \text{dla } x < 0 \\ 0 & \text{dla } x = 0 \\ 1 & \text{dla } x > 0 \end{cases}$$

Wskazówka: Spróbuj rozwiązać zadanie samodzielnie, a następnie przeczytaj §2.7 książki „Uczta programistów”.

Zadanie 9. Uzupełnij ciało funkcji zadeklarowanej poniżej.

```
/* Kiedy x zawiera nieparzystą liczbę jedynek zwróć 1, w p.p. 0 */
int32_t odd_ones(uint32_t x);
```

Wskazówka: Spróbuj rozwiązać zadanie samodzielnie, a następnie przeczytaj §5.2 książki „Uczta programistów”.