

Zad 2.

sobota, 1 kwietnia 2023 15:46

Zadanie 2. Zaimplementuj funkcję zdefiniowaną poniżej w asemblerze x86-64. Taka procedura w języku C miałaby sygnaturę «long cmp(uint64_t x, uint64_t y)».

$$\text{cmp}(x, y) = \begin{cases} -1 & \text{gdy } x < y \\ 1 & \text{gdy } x > y \\ 0 & \text{gdy } x = y \end{cases} \quad \begin{matrix} x - y < 0 \\ x - y > 0 \Rightarrow -(x - y) < 0 \end{matrix}$$

Wskazówka: Rozwiązanie wzorcowe ma cztery wiersze (bez ret). Użyj instrukcji adc, sbb i neg.

Handwritten assembly code and logic for the `cmp` function:

```

cmp:
    subq    %rsi, %rdi
    sbbq    %rax, %rax
    negq    %rdi
    adcq    %rax, %rax
    ret
    
```

Logic for the carry flag (CF) and the final result in `%rax`:

- Initial CF: $CF = \begin{cases} 1 & x < y \\ 0 & x \geq y \end{cases}$
- After `subq %rsi, %rdi`: $\%rax = \begin{cases} -1 & x < y \\ 0 & x \geq y \end{cases}, CF = 0$
- After `sbbq %rax, %rax`: $CF = \begin{cases} 0 & x = y \\ 1 & \text{wpp} \end{cases}$
- After `negq %rdi`: $\%rax = \begin{cases} 0 & x = y \\ -1 - 1 + 1 = -1 & x < y \\ 1 & x > y \end{cases}$
- After `adcq %rax, %rax`: (The final result is already in `%rax` after the previous instructions, so this instruction does not change the value.)