17 kwietnia 2023

Zadanie 5

Napisz i przetestuj następujący zestaw funkcji *rekurencyjnych* (w ich treści nie może być żadnych pętli; w implemetacji pierwszych trzech nawet instrukcje **if** nie powinny być potrzebne)

- `int gcdRec(int a, int b);` — zwraca największy wspólny dzielnik dwóch liczb naturalnych (całkowitych dodatnich);
- `int sumDigits(int n);` — zwraca sumę (dziesiętnych) cyfr podanej liczby naturalnej (e.g., 34→7);
- `int numDigits(int n);` — zwraca liczbę (dziesiętnych) cyfr podanej liczby naturalnej (e.g., 34→2);
- `void printOddEven(int n);` — drukuje w jednej linii
  - dla `n` parzystego – wszystkie liczby parzyste poczynając od 2 aż do `n`;
  - dla `n` nieparzystego – wszystkie liczby nieparzyste od 1 aż do `n`.

  (załóż, że `n` jest dodatnie);
- `void hailstone(int n);` — drukuje w jednej linii wszystkie liczby ciągu Collatza (patrz niżej) od `n` aż do 1.

Ciąg Collatza, znany też jako ciąg Ulama lub „gradowy” (ang. *hailstone*), to ciąg dla którego pierwszy wyraz a_0 jest dodatnią liczbą całkowitą, a kolejne wyrazy są wyliczane ze wzoru

$$a_{n+1} = \begin{cases} a_n/2 & \text{jeśli } a_n \text{ jest parzyste,} \\ 3a_n + 1 & \text{jeśli } a_n \text{ jest nieparzyste.} \end{cases}$$

Hipoteza Collatza (wciąż nieudowodniona) mówi, że niezależnie od wartości pierwszego elementu a_0, po skończonej liczbie kroków ciąg osiągnie wartość 1 (i potem będzie już cyklicznie przybierał wartości 1→4→2→1...).
Następujacy program, po zdefiniowaniu wszystkich funkcji

download *SimpleRekur.cpp*

```
#include <iostream>

int gcdRec(int a, int b);
int sumDigits(int n);
int numDigits(int n);
void printOddEven(int n);
void hailstone(int n);

int main() {
    std::cout << "gcdRec(12, 42) = " <<
                 gcdRec(12, 42) << std::endl
```

```
                  << "gcdRec(12, 25) = " <<
                      gcdRec(12, 25) << std::endl;
        std::cout << "sumDigits(123) = " <<
                      sumDigits(123) << std::endl
                  << "sumDigits(971) = " <<
                      sumDigits(971) << std::endl;
        std::cout << "numDigits(12345) = " <<
                      numDigits(12345) << std::endl
                  << "numDigits(971) = " <<
                      numDigits(971) << std::endl;
        std::cout << "printOddEven(15): ";
        printOddEven(15);
        std::cout << std::endl;
        std::cout << "printOddEven(14): ";
        printOddEven(14);
        std::cout << std::endl;
        std::cout << "hailstone(13): ";
        hailstone(13);
        std::cout << std::endl;
    }
```

powinien wypisać

```
gcdRec(12, 42) = 6
gcdRec(12, 25) = 1
sumDigits(123) = 6
sumDigits(971) = 17
numDigits(12345) = 5
numDigits(971) = 3
printOddEven(15): 1 3 5 7 9 11 13 15
printOddEven(14): 2 4 6 8 10 12 14
hailstone(13): 13 40 20 10 5 16 8 4 2 1
```

Nie używaj żadnych tablic, kolekcji ani zmiennych globalnych.

Termin: do 26 kwietnia (włącznie)

Rozwiązania, w postaci ***jednego*** *pliku źródłowego zawierającego treść programu, proszę wrzucać w systemie EDU do katalogu „Foldery zadań / Zadanie_XX", gdzie 'XX' jest numerem zadania.*

Nazwą pliku powinno być nazwisko z dużej litery (bez polskich znaków); rozszerzeniem musi być '`.cpp`*', czyli np.* `Malinowska.cpp`*.*