Problem Collatza

Problem Collatza (Collatz Conjecture), znana również jako problem 3n+1, to jedno z najbardziej znanych nierozwiązanych problemów w matematyce. Zakłada, że startując od dowolnej dodatniej liczby całkowitej, można ją zredukować do 1, wykonując iteracyjne kroki na podstawie następujących reguł: jeśli liczba jest parzysta, podziel ją przez 2; jeśli jest nieparzysta, pomnóż przez 3 i dodaj 1. Mimo prostoty zasady, problem pozostaje nierozwiązany dla dowolnej liczby startowej, choć jest uznawany za prawdziwy ze względu na obserwacje empiryczne dla wielu początkowych wartości.

Zadanie

1. Stwórz bibliotekę w języku C wystawiającą klientom następujące dwie funkcje:
    1. `int collatz_conjecture(int input)` - funkcja realizująca regułę Collatza postaci:

$$f(n) = \begin{cases} n/2 & \text{if } n \equiv 0 \pmod 2, \\ 3n+1 & \text{if } n \equiv 1 \pmod 2. \end{cases}$$

    Funkcja ta przyjmuje jedną liczbę typu całkowitego. Jeżeli jest ona parzysta, podziel ją przez 2 i zwróć wynik. Jeżeli jest nieparzysta, pomnóż ją przez 3 i dodaj 1, a następnie zwróć wynik.
    2. `int test_collatz_convergence(int input, int max_iter)` - funkcja sprawdzająca po ilu wywołaniach `collatz_conjecture` zbiega się do 1. Powinna ona wywoływać regułę Collatza najpierw na liczbie wejściowej a później na wyniku otrzymanym z reguły. W celu ochrony przed zbyt długim zapętlaniem się funkcji drugi parametr stanowi górną granicę liczby iteracji. W przypadku gdy funkcja wykona maksymalną ilość iteracji i nie znajdzie wyniku 1, wtedy zwróć -1.
2. W pliku makefile utwórz dwa wpisy: do kompilacji statycznej biblioteki i do kompilacji współdzielonej.
3. Napisz klienta korzystającego z kodu biblioteki, klient powinien sprawdzać kilka liczb, wykorzystując `test_collatz_convergence`, tj. po ilu iteracjach wynik zbiegnie się do 1 i wypisać liczbę iteracji na standardowe wyjście. Klient powinien korzystać z biblioteki na 3 sposoby:
    1. Jako biblioteka statyczna
    2. Jako biblioteka współdzielona (linkowana dynamicznie)
    3. Jako biblioteka ładowana dynamicznie

    Dla każdego z wariantów utwórz odpowiedni wpis w Makefile. Do realizacji biblioteki dynamicznej użyj definicji stałej (-D) i dyrektywy preprocesora, aby zmodyfikować sposób działania klienta.
4. Wyświetl zawartość plików binarnych wszystkich wariantów klienta przy pomocy programu *objdump*, znajdź miejsca wywołania funkcji `test_collatz_convergence` omów różnice w kodzie binarnym. Dla większej jasności kodu kompiluj bez optymalizacji -O0.