*W wyniku wykonania poniższych zadań powinien powstać **program**, którego kody źródłowe powinny zostać przesłane na adres antyplagiatu (informacja w ramce na końcu zadania).*

Naszym zadaniem będzie napisanie programu, który pozwoli na przetestowanie obsługi przychodzących sygnałów (zwykłych i czasu rzeczywistego).

Program uruchamiamy z dwoma parametrami liczbowymi:
- maksymalną długością życia procesów potomnych (sekundy) oraz
- przerwą pomiędzy ich tworzeniem (sekundy).

Program główny w pętli, w stałych odstępach czasu (podanych jako argument wywołania), tworzy nowy proces potomny. Proces potomny losuje liczbę z zakresu do podanego jako argument maksimum. Następnie wykonuje dowolne obliczenia (np. liczymy kolejne wartości silni) przez okres równy wylosowanej wartości (posłużyć się funkcją **alarm**). Proces potomny powinien obsłużyć sygnał **SIGALRM** (patrz f-cja alarm), wysłać do rodzica sygnał czasu rzeczywistego z wcześniej wylosowaną liczbę i zakończyć się.

Posłużenie się sygnałami czasu rzeczywistego pozwoli na ich kolejkowanie po stronie rodzica, gdyby jednocześnie kilka procesów potomnych w tym samym czasie kończyło działanie.

Każdy proces potomny po uruchomieniu wyświetla informację o sobie w jednej kolumnie (pid, wylosowana wartość, czas utworzenia). Główny program aktywnie czeka na sygnały od dzieci wysyłane przed ich zakończeniem: po odebraniu sygnału od potomka wyświetla w drugiej kolumnie otrzymane informacja (pid procesu potomnego, wylosowaną w procesie potomnym wartość, czas zakończenia).

Podpowiedź: Należy użyć funkcji **sigaction**, zwracając szczególną uwagę na flagę **SA_SIGINFO** i budowę struktury **siginfo_t**.

Program działa w pętli tak długo, aż otrzyma sygnał **SIGINT**. Powinien wtedy zaczekać na zakończenie wszystkich uruchomionych do tej pory potomków, nie tworząc już nowych i zakończyć swoje działanie. Zwróć uwagę, żeby **SIGINT** wysyłane za pomocą kombinacji klawiszy **Ctrl-C** nie przerwało działania procesów potomnych (sygnał kierowany jest w takiej sytuacji do grupy procesów).

Przykładowy wynik działania programu (procesy potomne tworzone w 10 sekundowych odstępach, maksymalna długość działania procesu potomnego 20 sekund).

```
./testSig -w 10 -m 20

[ 1051 ]  [ 3 ]  [ Fri Mar 19 19:30:11 2021 ]
```

```
                              [ 1051 ]  [ 3 ] [ Fri Mar 19 19:30:14 2021 ]
[ 1052 ]  [ 12 ]  [ Fri Mar 19 19:30:21 2021 ]
[ 1053 ]  [ 16 ]  [ Fri Mar 19 19:30:31 2021 ]
^C
                              [ 1052 ]  [ 12 ] [ Fri Mar 19 19:30:34 2021 ]
                              [ 1053 ]  [ 16 ] [ Fri Mar 19 19:30:47 2021 ]
```

Przed wysłaniem pliku źródłowego na antyplagiat jego nazwę zmieniamy na: ***numer_indeksu.ps.lab05.main.c*** *(czyli np. 66666.ps.lab05.main.c).*

Kody źródłowy (1 plik) po oddaniu prowadzącemu zajęcia laboratoryjne muszą zostać jako załączniki przesłane na adres **pss1@zut.edu.pl** (wysyłamy jeden mail z czterema załącznikami):

- pliki z kodami źródłowymi muszą mieć nazwę zgodne ze wzorcem podanym w treści zadania,
- mail musi zostać wysłany z poczty uczelnianej (domena **zut.edu.pl**),
- temat maila musi mieć postać: **PS IS1 999X LAB05**, gdzie 999X to numer grupy laboratoryjnej (np. PS IS1 321 LAB05),
- w pierwszych trzech liniach kodu źródłowego w komentarzach (każda linia komentowana osobno) musi znaleźć się:
  - informacja identyczna z zamieszczoną w temacie maila,
  - imię i nazwisko osoby wysyłającej maila,
  - adres e-mail, z którego wysłano wiadomość,
    np.:

    // PS IS1 321 LAB05
    // Jan Nowak
    // nj66666@zut.edu.pl

- e-mail nie może zawierać żadnej treści (tylko załączniki).

Dostarczone kody programów będą analizowane pod kątem wykrywania plagiatów. Niewysłanie wiadomości, wysłanie jej w formie niezgodnej z powyższymi wymaganiami lub wysłanie pliku, który nie będzie się kompilował i uruchamiał, będzie traktowane jako brak programu i skutkowało otrzymaniem za niego oceny niedostatecznej.