Temat zajęć	Procesy w systemie Windows
Zakres materiału	Tworzenie i wykonywanie procesów w systemie Windows

Materiał teoretyczny

- funkcje *CreateProcessA(), WaitForSingleObject()*, *WaitForMultipleObject(), GetExitCodeProcess()*
- analiza argumentów linii poleceń i kodu zakończenia procesu

Treść zadania

Napisać języku C program spełniający poniższe wymagania:

- program akceptuje dokładnie jeden argument wywołania i jest nim ciąg cyfr dziesiętnych o długości od 1 do 25; uruchomienie programu z inną liczbą argumentów, albo bez argumentów, albo z argumentem dłuższym niż 25 znaków, albo z argumentem, który zawiera inne znaki niż cyfry dziesiętne powinno skutkować wyprowadzeniem na *stderr* informacji o błędzie i zakończeniem pracy z kodem zakończenia równym 1;
- jeżeli otrzymany argument ma długość 1, program kończy pracę, zwracając jako kod zakończenia wartość otrzymanej cyfry (np. dla `'1'` zwraca `1`, itd).
- w przeciwnym przypadku program dzieli otrzymany w argumencie łańcuch na dwie „połowy” (jeśli długość łańcucha nie jest parzysta, druga z „połówek” będzie dłuższa o jeden znak);
- program uruchamia sam siebie w dwóch procesach potomnych (funkcja *CreateProcessA()*);
- każdy z procesów potomnych otrzymuje jako argument odpowiednio: pierwszy – pierwszą część łańcucha argumentu, drugi – drugą część argumentu;
- rodzic czeka na zakończenie obu procesów potomnych (funkcja *WaitForSingleObject()* albo *WaitForMultipleObject()*), a następnie dla każdego z dzieci wypisuje na *stdout*: swój PID, PID zakończonego dziecka, argument, z którym dziecko zostało uruchomione i kod zakończenia dziecka;
- rodzic kończy pracę, zwracając kod powrotu równy sumie kodów powrotu otrzymanych od dzieci.

Uwaga: poprawna sekwencja operacji to *Create-Create-Wait-Wait* (czyli rodzic **najpierw** uruchamia **dwa** procesy potomne, a potem czeka na zakończenie obu); konstrukcja kodu, w której procesy potomne pracują po kolei (*Create-Wait-Create-Wait*) dyskwalifikuje rozwiązanie!

- przykładowe wyjście z programu:

```
C:\> lab9 98765
  1468   2164                 9 9
  1468   3968                 8 8
  1556   1468                98 17
  3608   2928                 7 7
  4024   3008                 6 6
  4024   3476                 5 5
  3608   4024                65 11
  1556   3608               765 18
```

Uwaga! Kod źródłowy programu (1 plik) po oddaniu prowadzącemu zajęcia laboratoryjne musi zostać jako **załącznik** przesłany na adres `sos1@wi.zut.edu.pl`:

- plik z kodem źródłowym musi mieć nazwę: `numer_indeksu.so.lab09.c` (np. `66666.so.lab09.c`),
- plik musi zostać wysłany z poczty uczelnianej (domena `zut.edu.pl`),
- temat maila musi mieć postać: `SO IS1 999X LAB09`
  gdzie *999X* to numer grupy laboratoryjnej (np. `SO IS1 210C LAB09`),
  - w pierwszych trzech liniach skryptu 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.:
    ```
    // SO IS1 210C LAB09
    // Jan Nowak
    // nj66666@zut.edu.pl
    ```
  - e-mail nie może zawierać żadnej treści (tylko załącznik).

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