|  |  |  |
| --- | --- | --- |
| 16.10.2023 | Łukasz Oprych  Gr.5 ITE | Temat:  POSIX |

Cel:

Zapoznanie się z operacjami na wątkach przy pomocy biblioteki pthreads.

Przebieg ćwiczenia:

Uzupełnienie skryptu zgodnie z poleceniem prowadzącego:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Za pomocą funkcji pthread\_create utworzono nowy wątek, przekazując NULL jako argument attr, co oznacza, że przyjęto "domyślne wartości" dla tego nowego wątku. ID tego nowego wątku w systemie zostało zapisane do zmiennej tid. Następnie, przy użyciu funkcji pthread\_join, czekało się aktywnie na zakończenie działania tego wątku. Wynik zwracany przez ten wątek został zapisany w zmiennej wynik.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Utworzono nowy wątek i odłączono go od wątku głównego za pomocą pthread\_detach. Następnie, w pętli, co każdą iterację, wątek główny był zawieszany na 1 sekundę, co ostatecznie sumowało się do łącznego oczekiwania przez 10 sekund na zakończenie pracy tego oddzielonego wątku. Zatrzymywano ten proces, gdy "zmienna\_wspolna" przyjmowała wartość jeden. To oznaczało, że wątek nie odebrał sygnału zatrzymania i zakończył swoje działanie bez wymuszenia. Po zakończeniu pętli dodawano informację, która na podstawie wartości "zmienna\_wspolna" określała, czy wątek został przerwany, czy nie. Wartość równa 0 wskazywała na przerwanie wątku, podczas gdy Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatyczniewartość różna od zera sugerowała, że wątek nie otrzymał sygnału zatrzymania.

Następnie zaczęto od inicjalizacji domyślnych ustawień atrybutów wątku w zmiennej "attr" i ustawiono atrybut, który odpowiada za stworzenie wątku jako "odłączony" podczas jego tworzenia. Następnie przekazano zmienną z tymi atrybutami jako drugi argument do funkcji pthread\_create, co skutkowało stworzeniem wątku, który był już odłączony od wątku głównego od samego początku. Aby utrzymać spójność, zniszczono zmienną, która przechowywała te atrybuty, gdy już nie były potrzebne.

Następnie uruchomiono program:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Następnie utworzono katalog zad\_2 wraz z nowym plikiem źródłowym.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

W funkcji „zadanie\_watku” uzyskiwany jest identyfikator danego wątku, przy użyciu pthread\_self i przechowywany w zmiennej x. Na koniec funkcja wypisuje przechowywany numer wątku oraz indetyfikator systemowy wątku.

Liczba wątków została określona za pomocą makra preprocesora, co umożliwia łatwą modyfikację liczby elementów w tablicy „indeksy”. Przypisywaliśmy kolejne wartości do poszczególnych komórek w tej tablicy, zaczynając od 0 i kontynuując aż do wartości o jeden mniejszej niż LICZBA\_W\_MAX. Dopiero po wykonaniu tego procesu tworzono wątki, a następnie oczekiwano, aż wszystkie zakończą swoje działanie. Na koniec używając pthread\_exit kończymy działanie wątku głównego.

Wynik programu:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Wnioski:

Wątki, mogą pracować w trybie detached, czyli odłączonym oraz joinable – standardowym. Wątek standardowy, wykonuje swoje polecenia i zakańcza po tym swoje działanie, można też to wymusić za pomocą funkcji pthread\_cancel. W celu zabezpieczenia wątku standardowego przed zakończeniem jego działaniem możemy zastosować funkcję pthread\_cancel\_disable, co blokuje wykonywanie funkcji pthread\_cancel. Tworzenie wątków odłączonych jest praktyczne w sytuacjach, gdy nie mamy zamiaru oczekiwać na ich zakończenie za pomocą funkcji pthread\_join. Wątek odłączony samodzielnie zwalnia zasoby po swoim zakończeniu, podczas gdy wątek nieodłączony wymaga, aby inny wątek "dołączył" do niego, aby te zasoby zostały zwolnione.