

Systemy operacyjne

Sprawozdanie - laboratorium 5 „Procesy 2”

Andrzej Kołakowski
296586

1) Tworzenie wątku

1. Różnice między procesem a wątkiem

Proces to egzemplarz wykonywanego programu. Każda aplikacja może składać się z jednego lub więcej procesów.

Wątek to część programu wykonywana współbieżnie w obrębie jednego procesu. W jednym procesie może istnieć wiele wątków.

Różnica między procesem a wątkiem polega na współdzieleniu przez wszystkie wątki działające w danym procesie przestrzeni adresowej oraz wszystkich innych struktur systemowych (np. listy otwartych plików, gniazd itp.), z kolei procesy posiadają niezależne zasoby.

2. Funkcja `pthread_create`

Funkcja ta służy do tworzenia nowego wątku.

Jako argumenty przyjmuje bufor, w którym zostanie zapisane ID nowego wątku, strukturę w której zapisane są atrybuty nowo-tworzonego wątku, funkcję która zostanie uruchomiona oraz jej argument.

W przypadku pomyślnego zakończenia zwraca 0, w innym wypadku zwraca kod błędu.

3. Modyfikacja programu `hello.c` tak, aby tworzył 10 wątków, z których każdy wypisze swój numer przesłany jako argument funkcji rozpoczęcia

```
root@localhost:~/Desktop/so/lab5
File Edit View Search Terminal Help
My thread id: 8598
My number: 4
My thread id: 8599
My number: 5
My thread id: 8600
My number: 6
My thread id: 8601
My number: 7
My thread id: 8602
My number: 8
My thread id: 8603
My number: 9
My thread id: 8604
My number: 10
My thread id: 8605
[root@localhost lab5]#
```

hello.c

2) Czekanie na zakończenie wątku

1. Funkcja `pthread_join`

Funkcja ta czeka na zakończenie wątku wskazanego w argumencie.

W przypadku pomyślnego zakończenia zwraca 0, w innym wypadku zwraca kod błędu.

2. Program `hello.c` zmodyfikowany w taki sposób, aby wątek główny czekał na zakończenie pracy przez pozostałe wątki.

```
root@localhost:~/Desktop/so/lab5
File Edit View Search Terminal Help

My number: 10
My thread id: 8779

My number: 7
My thread id: 8776

My number: 9
My thread id: 8778

My number: 2
My thread id: 8771

My number: 4
My thread id: 8773

My number: 8
My thread id: 8777

My number: 6
My thread id: 8775

End of the main thread!
[root@localhost lab5]#
```

hello2.c

3) Synchronizacja wątków (muteksy)

1. Funkcje powiązane z pracą na muteksach

`pthread_mutex_lock` - funkcja ta blokuje obiekt mutex wskazany w argumencie. Jeżeli obiekt jest już zablokowany przez inny wątek, to funkcja oczekuje na jego zwolnienie.

`pthread_mutex_unlock` - funkcja ta odblokowuje obiekt mutex wskazany w argumencie.

`pthread_mutex_init` - funkcja ta inicjalizuje obiekt mutex podanymi atrybutami. Jeżeli jako argument podane zostanie NULL, używane są domyślne atrybuty.

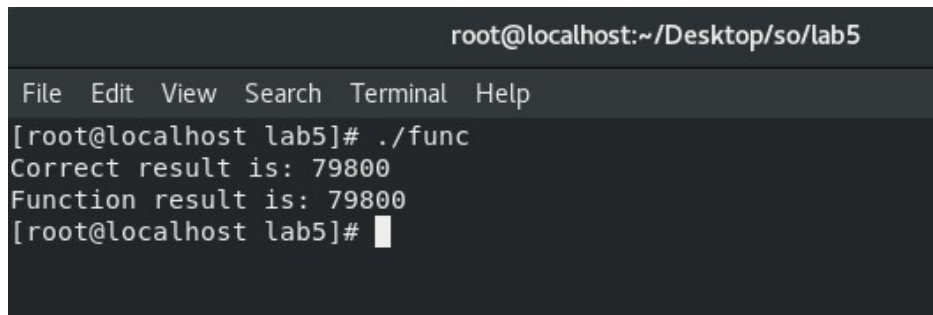
`pthread_mutex_destroy` - funkcja ta niszczy obiekt mutex wskazany w argumencie.

W przypadku pomyślnego zakończenia wszystkie funkcje zwracają 0, w innym wypadku zwracają kody błędów.

2. Program `func.c`

- Realizuje sumowanie wartości w tablicy przy pomocy wątków

- Wątek główny wypełnia tablicę znajdującą się w globalnej instancji struktury `CommonData`.
- Tworzy `NUM` nowych wątków, których zadaniem jest obliczenie sumy poszczególnych fragmentów tablicy.
- Wątki zapisują obliczone przez siebie wyniki (częstkowe) do pola `sum` zmiennej `CommonData`, do której dostęp jest synchronizowany mutexami.



```

root@localhost:~/Desktop/so/lab5
File Edit View Search Terminal Help
[root@localhost lab5]# ./func
Correct result is: 79800
Function result is: 79800
[root@localhost lab5]#

```

func.c

4) Zmienne warunkowe

1. Wybrane funkcje

`pthread_cond_signal` - funkcja ta odblokowuje wątek zablokowany na zmiennej warunkowej.

`pthread_cond_wait` - funkcja ta zwalnia obiekt mutex i blokuje wątek na zmiennej warunkowej.

`pthread_cond_init` - funkcja ta inicjalizuje obiekt cond podanymi atrybutami. Jeżeli jako argument podane zostanie `NULL`, używane są domyślne atrybuty.

`pthread_cond_destroy` - funkcja ta niszczy obiekt cond wskazany w argumencie.

W przypadku pomyślnego zakończenia wszystkie funkcje zwracają 0, w innym wypadku zwracają kody błędów.

2. Program `cond.c`

- 2 wątki inkrementują (funkcja `increment`) wartość zmiennej globalnej `globalvariable`
- Trzeci wątek (funkcja `printinfo`) oczekuje na sygnał, aby oznajmić, że osiągnięto żadaną wartość `MAXVAL`.
- Po wypisaniu informacji wszystkie wątki i cały program kończą działanie.

```
root@localhost:~/Desktop/so/lab5
File Edit View Search Terminal Help
[root@localhost lab5]# ./cond
Osiagnieto zadana wartosc MAXVAL= 100
t1 finished!
t2 finished!
t3 finished!
Finishing...
[root@localhost lab5]#
```

cond.c

5) Kasowanie wątku

1. Przegląd funkcji

`pthread_exit` - funkcja ta kończy działanie wątku, który ją wywołał oraz zwraca kod wyjścia w swoim argumencie, który może być odczytany przez proces który wywoła `pthread_join`. Brak wartości zwracanej, zawsze zakończy się pomyślnie.

`pthread_cancel` - funkcja ta wysyła żądanie zatrzymania wykonywania do wątku podanego w argumencie.

`pthread_testcancel` - funkcja ta tworzy punkt w wątku który ją wywołał, w którym może obsłużyć żądanie zatrzymania wykonywania.

`pthread_cleanup_push` - funkcja ta dodaje na stos procedurę oraz jej argument, która zostanie automatycznie wywołana gdy wątek zostanie zatrzymany.

`pthread_cleanup_pop` - funkcja ta zdejmuje ze stosu procedurę i w przypadku niezerowego argumentu wykonuje ją.

2. Program `randomsearch.c`

Uzupełniono program o przesłanie do tworzonych wątków argumentów zawierających informacje o indeksie wątku oraz wartości szukanej i wyświetlanie informacji o liczbie iteracji, jakie wykonał każdy kończący się wątek.

```
root@localhost:~/Desktop/so/lab5
File Edit View Search Terminal Help
[root@localhost lab5]# ./randomsearch
Searching for: 100
Number found by 2!
Thread 1 tries count= 0
Thread 4 tries count= 922
Thread 0 tries count= 85
Thread 3 tries count= 0
Thread 2 tries count= 14
Number of all iterations: 1021.
```

randomsearch.c

6) Zadanie dodatkowe

Program złożony z 3 wątków:

- Wątek główny losowo generuje elementy tablicy `int tab[2][10]`.
- Wątek 2. liczy sumę elementów pierwszego wiersza tablicy.
- Wątek 3. liczy sumę elementów drugiego wiersza tablicy.
- Wątek główny liczy sumę całkowitą z sum częściowych wyznaczonych przez wątki 2. i 3 sumując rezultaty zwrócone przez wątki.

Ilość wierszy i kolumn tablicy zdefiniowałem używając `#define` dzięki temu łatwo zmodyfikować program do dowolnych wymiarów tablicy i ilości tworzonych wątków.

```
root@localhost:~/Desktop/so/lab5
File Edit View Search Terminal Help
[root@localhost lab5]# ./6
Sum from thread 0 = 5511
Sum from thread 1 = 5123
Correct result is: 10634
Sum from threads is: 10634
[root@localhost lab5]#
[root@localhost lab5]# ./6
Sum from thread 1 = 3776
Sum from thread 0 = 5591
Correct result is: 9367
Sum from threads is: 9367
[root@localhost lab5]# █
```