Tomasz Ligęza

Programowanie równoległe. Przetwarzanie równoległe i rozproszone.

Sprawozdanie z laboratorium 3.

Cel zajęć:

- nabycie praktycznej umiejętności manipulowania wątkami Pthreads – tworzenia, niszczenia, elementarnej synchronizacji,
- przetestowanie mechanizmu przesyłania argumentów do wątku,
- poznanie funkcjonowania obiektów określających atrybuty wątków.

W ramach zajęć zrealizowałem następujące kroki:

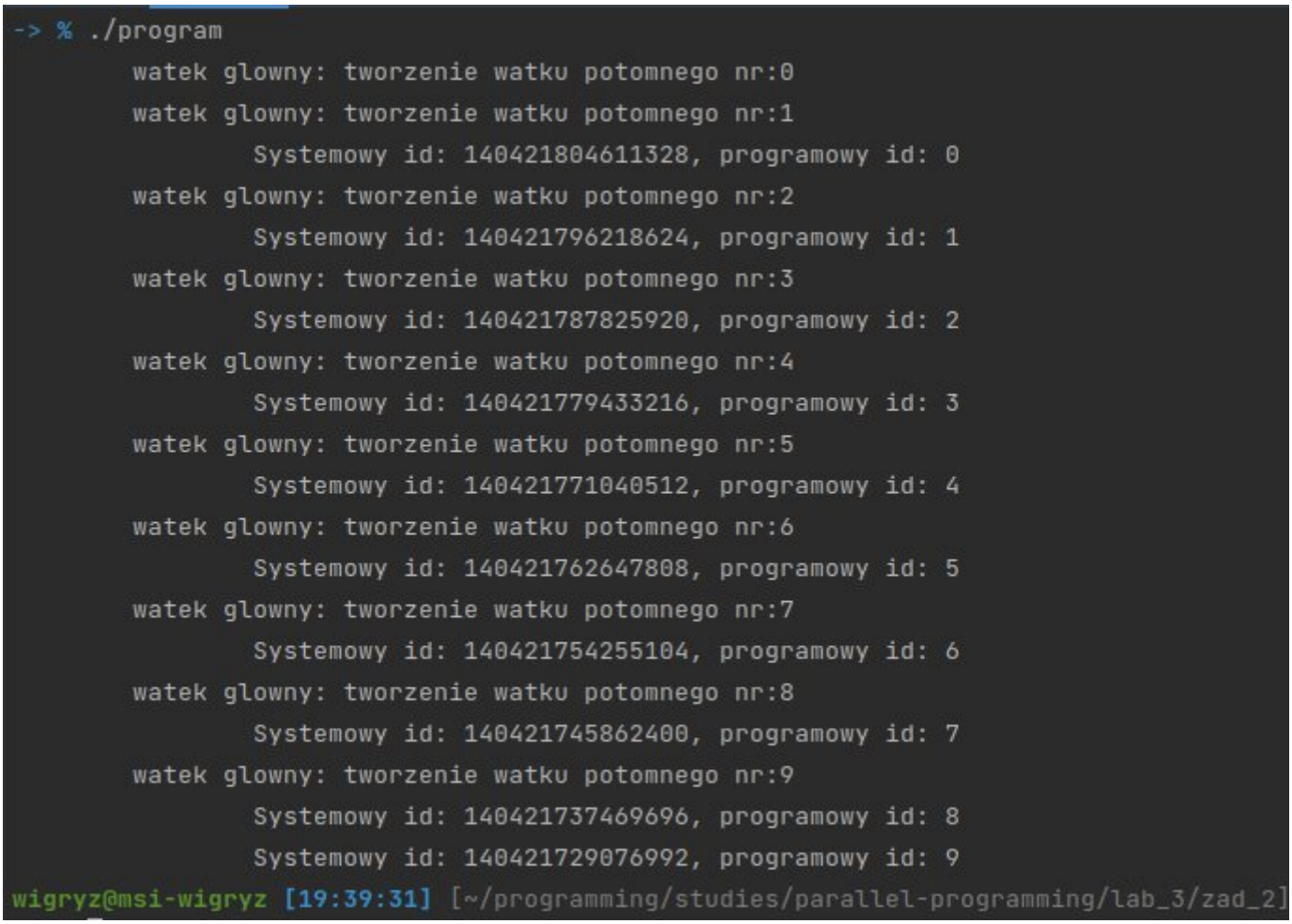
1. Uzupełniłem kod programu pthreads_detach_kill.c

```
/*Tu wstaw kod tworzenia wątku z domyślnymi własnościami*/
  i = pthread_create(&amp;tid, NULL, zadanie_watku, NULL);
//Co nalezy zrobić przed sprawdzeniem czy wątki się skonczyły?
  i = pthread_join(tid, &amp;wynik);
/*Tu wstaw kod tworzenia wątku z domyślnymi własnościami*/
  i = pthread_create(&amp;tid, NULL, zadanie_watku, NULL);
 pthread_detach(tid);

//Czy watek został zabity? Jak to sprawdzić?
for(int i=0 ; i < 5 ; i++) {
      sleep(1);
      if(zmienna_wspolna != 0)
         break;
}

if(zmienna_wspolna == 0) {
      printf("ZMIENNA NIE ZOSTAŁA ZMIENIONA. PRAWDOPODOBNIE ZABITY\n");
}
else {
      printf("ZMIENNA ZOSTAŁA ZMIENIONA. PRZEZYL\n");
}
//Inicjacja atrybutów?
pthread_attr_init(&amp;attr);
//Ustawianie typu watku na odłaczony
pthread_attr_setdetachstate(&amp;attr, PTHREAD_CREATE_DETACHED);
pthread_create(&amp;tid, &amp;attr, zadanie_watku, NULL);
pthread_attr_destroy(&amp;attr);

printf("\twatek glowny: koniec pracy, watek odlaczony pracuje dalej\n");
pthread_exit(NULL); // co stanie sie gdy uzyjemy exit(0)?
//exit(0); TO ZABIJA PROCES MAIN I POTOMNY
```

2. Utworzyłem nowy program w odrębnym podkatalogu:

```
void * zadanie_watku (void * arg_wsk) {
   int id = *((int *) arg_wsk);
   printf("\t\tSystemowy id: %lu, programowy id: %d \n", pthread_self(), id);
   return (NULL);
}

int main()
{
   pthread_t tid[LICZBA_WATKOW];
   int ids[LICZBA_WATKOW];
   int tab[LICZBA_WATKOW];
   int wyniki[LICZBA_WATKOW];
   for(int i=0 ; i < LICZBA_WATKOW ; i++) {
      ids[i] = i;
   }
   for(int i=0 ; i < LICZBA_WATKOW ; i++) {
      printf("\twatek glowny: tworzenie watku potomnego nr:%d\n", i);
         //pthread_create(&(tid[i]), NULL, zadanie_watku, &(ids[i]));
      pthread_create(&(tid[i]), NULL, zadanie_watku, &i);
      // tutaj jesli przesylam wartosc &i to moze pojawic sie blad synchronizacji
   }
   for(int i=0 ; i < LICZBA_WATKOW ; i++) {
      tab[i] = pthread_join(tid[i], &(wyniki[i]));
   }
  pthread_exit(NULL);
}
```

```
                Systemowy id: 140032801654528, programowy id: 6
        watek glowny: tworzenie watku potomnego nr:7
        watek glowny: tworzenie watku potomnego nr:8
        watek glowny: tworzenie watku potomnego nr:9
                Systemowy id: 140032776476416, programowy id: 9
                Systemowy id: 140032793261824, programowy id: 9
                Systemowy id: 140032768083712, programowy id: 10
                Systemowy id: 140032784869120, programowy id: 10
wigryz@msi-wigryz [19:24:01] [~/programming/studies/parallel-programming/lab_3/zad_2]
```



Odpowiedzi na pytania z instrukcji:

1. W jakich dwóch trybach mogą funkcjonować wątki Pthreads? Jaka jest różnica między tymi trybami?

Wątki pthreads mogą działać w trybie odłączonym (detached) i standardowym.
Wątku w trybie odłączonym nie możemy dołączyć poprzez pthread_join.

2. Kiedy wątek standardowo kończy swoje działanie?

Wątek kończy swoje działanie w momencie zakończenia wszystkich wykonywanych instrukcji.

3. W jaki sposób można wymusić zakończenie działania wątku? (czym różnią się w tym przypadku wątki odłączone i standardowe?)

Wymusić zakończenie pracy wątku możemy poprzez ich jego zabicie - poprzez wywołanie metody „pthread_cancel”.

4. Jak wątek może chronić się przed próbą "zabicia"?

Wątek może stać się odporny na próby zabicia poprzez wywołanie funkcji:
pthread_setcancelstate(PTHREAD_CANCEL_DISABLE, _);

5. Jak można sprawdzić czy próba "zabicia" wątku powiodła się? (czym różnią się w tym przypadku wątki odłączone i standardowe?)

Możemy to sprawdzić poprzez wywołanie pthread_join(tid_wątku, &wynik) i sprawdzenie czy wynik == PTHREAD_CANCEL.
W przypadku wątku odłączonego nie możemy tego zrobić, ale możemy to obejść np. sprawdzeniem czy zmienna globalna, która miała być przez wątek modyfikowana zmieniła swoją wartość.

6. Jaki może pojawić się błąd synchronizacji w przypadku próby przesłania zwykłego wskaźnika do liczby całkowitej?

Zmienna zostanie zmodyfikowana poza procesem, który otrzymał zmienną.

Wnioski:

Przesyłanie zmiennych do wątku może prowadzić do tzw. race condition. Oznacza to w skrócie tyle, że w tym samym czasie dwa wątki operują na tej samej zmiennej, przez co generowane często trudne do wytropienia błędy.