

EMILIA STARCZYK
249005

LISTA 7 - ZADANIA 5,6,7
Systemy Czasu Rzeczywistego
MGR. INŻ. ARKADIUSZ MIELCZAREK
14 stycznia 2021

1. Zadanie do wykonania

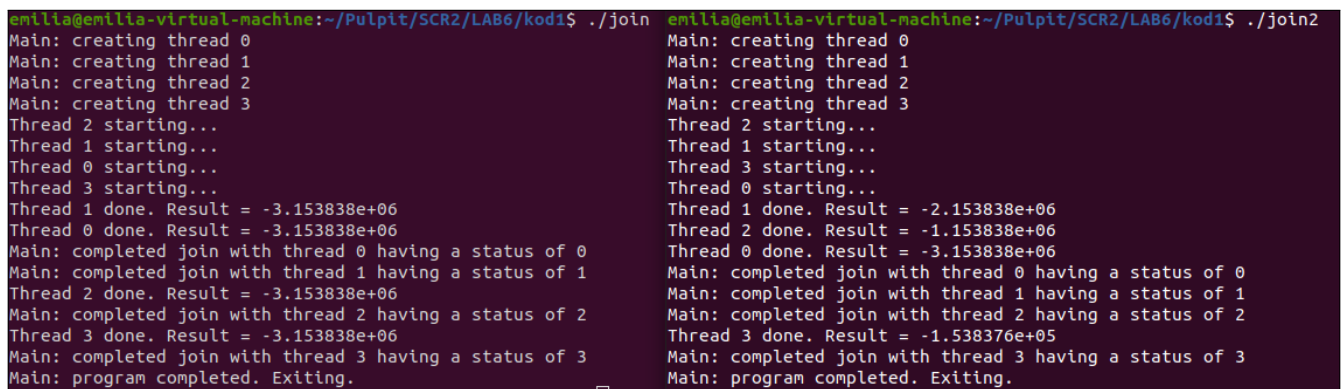
Należało wykonać zadania 5,6,7. Do zadania 7 należało napisać program w języku C.

1.1. Zadanie 5

Zadanie polegało na zrozumieniu kodu z pliku join.c, późniejszym zmodyfikowaniu go i zabserwowaniu różnic między programem join.c a programem detached.c. Program join.c wykorzystuje tworzenie wątków z atrybutem. Atrybutem tym jest domyślne PTHREAD_CREATE_JOINABLE. Tak naprawdę linii związanych z tym atrybutem mogłoby w programie nie być. Poprzez wykorzystanie funkcji pthread_join() w programie oczekujemy na zakończenie wszystkich wątków. Późniejsza część programu wykonuje się dopiero po tym fakcie. Oprócz zrozumienia programu wymagane było zmodyfikowanie programu tak, aby zwracał inne wartości. Zatem w linii 26 dopisałam dodanie wartości numeru wątku do wartości obliczanej.

```
26      result = result + sin(i) * tan(i) + tid;
```

Wyniki prezentują się następująco:



```
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB6/kod1$ ./join
Main: creating thread 0
Main: creating thread 1
Main: creating thread 2
Main: creating thread 3
Thread 2 starting...
Thread 1 starting...
Thread 0 starting...
Thread 3 starting...
Thread 1 done. Result = -3.153838e+06
Thread 0 done. Result = -3.153838e+06
Main: completed join with thread 0 having a status of 0
Main: completed join with thread 1 having a status of 1
Thread 2 done. Result = -3.153838e+06
Main: completed join with thread 2 having a status of 2
Thread 3 done. Result = -3.153838e+06
Main: completed join with thread 3 having a status of 3
Main: program completed. Exiting.

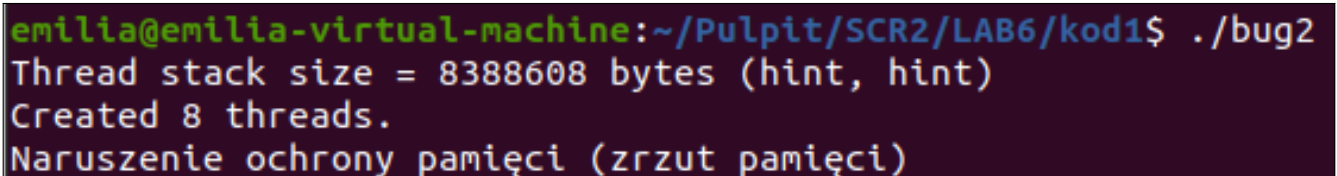
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB6/kod1$ ./join2
Main: creating thread 0
Main: creating thread 1
Main: creating thread 2
Main: creating thread 3
Thread 2 starting...
Thread 1 starting...
Thread 3 starting...
Thread 0 starting...
Thread 1 done. Result = -2.153838e+06
Thread 2 done. Result = -1.153838e+06
Thread 0 done. Result = -3.153838e+06
Main: completed join with thread 0 having a status of 0
Main: completed join with thread 1 having a status of 1
Main: completed join with thread 2 having a status of 2
Thread 3 done. Result = -1.538376e+05
Main: completed join with thread 3 having a status of 3
Main: program completed. Exiting.
```

Rysunek 1: Wartości obliczone przez wątki w programach join.c oraz join2.c

Program detached.c wykorzystuje atrybut wątku PTHREAD_CREATE_DETACHED. Takie wątki po zakończeniu swojego działania zwracają swoje zasoby do programu, tak aby mogły zostać ponownie użyte. Utworzenie wątku z tym atrybutem powoduje, że program główny może się wykonywać do końca bez obowiązku czekania na konkretny moment ukończenia się wątków - może sam wykonywać swoje zadanie. Program ostatecznie zakończy się przy ukończeniu pracy wszystkich wątków w tle.

1.2. Zadanie 6

Zadanie 6 polegało na zrozumieniu programu bug2.c i problemu w nim występującego. Program ma za zadanie wykonać pracę obliczeniową i wyświetlić napis na terminalu z wyznaczoną liczbą. Wątki mają domyślną wartość stosu. Zaraz po uruchomieniu programu wyrzucany jest błąd naruszenia ochrony pamięci.



```
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB6/kod1$ ./bug2
Thread stack size = 8388608 bytes (hint, hint)
Created 8 threads.
Naruszenie ochrony pamięci (zrzut pamięci)
```

Rysunek 2: Błąd naruszenia ochrony pamięci w programie bug2.c

Wątek próbuje się odwołać do pamięci, której nie ma przydzielonej do swojego użytku. Aby poprawić ten błąd należy wyznaczyć wartość stosu jaka jest potrzebna do wykonania zadania i przydzielić mu odpowiednią pamięć.

```

stacksize = ARRAY_SIZE * sizeof(double) + sizeof(double) + sizeof(int) +
            sizeof(long) + 10000;
pthread_attr_setstacksize(&attr, stacksize);

```

Aby utworzyć wątek z tym atrybutem należało w funkcji pthread_create() zamiast NULL umieścić odpowiedni atrybut. Przydzielenie odpowiedniej pamięci stosu powoduje, że program zaczyna działać poprawnie. Program bug2fix.c w bardzo podobny sposób został poprawiony.

1.3. Zadanie 7

Zadanie 7 polegało na napisaniu programu, który za pomocą pewnej liczby wątków będzie wyznaczał wartości liczby pi. Zgodnie z metodą Monte Carlo wartość liczby pi można wyznaczyć za pomocą stosunku pola koła wpisanego w kwadrat oraz pola samego kwadratu. Zakładając, że generujemy N-punktów, które znajdują się w kwadracie, sprawdzamy tylko dodatkowo czy znajdują się one w kole. Wartość pi wyliczana jest wtedy na podstawie wzoru:

$$\pi = \frac{4 * \text{liczba_punktów_w_kole}}{\text{liczba_wygenerowanych_punktów}}$$

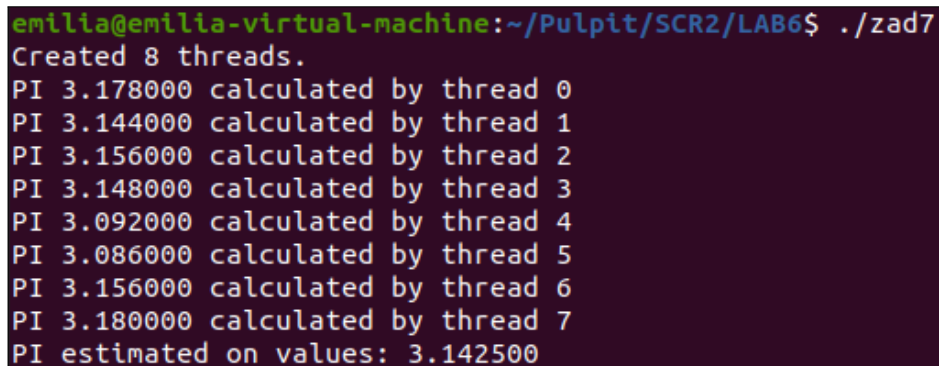
Poniżej znajduje się funkcja pojedynczego wątku, wyznaczająca wartość liczby pi.

```

void *thread_func(void *thread_arg)
{
    double point_x = 0;
    double point_y = 0;
    int points_in_circle = 0;
    thread_data *pi = (thread_data *)thread_arg;
    for (int i = 0; i < NPOINTS; i++)
    {
        //all points are inside square
        point_x = drand48() * RADIUS; //drand48() points between <0,1)
        point_y = drand48() * RADIUS;
        if (sqrt(pow(point_x, 2) + pow(point_y, 2)) <= RADIUS)
        { //check if point is inside circle
            points_in_circle++;
        }
    }
    pi->pi_value = ((4.0 * points_in_circle) / NPOINTS);
    pthread_exit(NULL);
}

```

Cały kod programu zawarty jest w pliku zad7.c. Wartość ilości wątków oraz liczby punktów dobrałam na podstawie testów.



```

emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB6$ ./zad7
Created 8 threads.
PI 3.178000 calculated by thread 0
PI 3.144000 calculated by thread 1
PI 3.156000 calculated by thread 2
PI 3.148000 calculated by thread 3
PI 3.092000 calculated by thread 4
PI 3.086000 calculated by thread 5
PI 3.156000 calculated by thread 6
PI 3.180000 calculated by thread 7
PI estimated on values: 3.142500

```

Rysunek 3: Wartości obliczone przez każdy z 8 wątków na podstawie 2000 punktów

```
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB6$ ./zad7
Created 8 threads.
PI 3.173333 calculated by thread 0
PI 3.085333 calculated by thread 1
PI 3.148000 calculated by thread 2
PI 3.133333 calculated by thread 3
PI 3.094667 calculated by thread 4
PI 3.120000 calculated by thread 5
PI 3.096000 calculated by thread 6
PI 3.161333 calculated by thread 7
PI estimated on values: 3.126500
```

Rysunek 4: Wartości obliczone przez każdy z 8 wątków na podstawie 3000 punktów

```
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB6$ ./zad7
Created 20 threads.
PI 3.178000 calculated by thread 0
PI 3.210000 calculated by thread 1
PI 3.102000 calculated by thread 2
PI 3.196000 calculated by thread 3
PI 3.154000 calculated by thread 4
PI 3.120000 calculated by thread 5
PI 3.094000 calculated by thread 6
PI 3.122000 calculated by thread 7
PI 3.102000 calculated by thread 8
PI 3.100000 calculated by thread 9
PI 3.156000 calculated by thread 10
PI 3.148000 calculated by thread 11
PI 3.108000 calculated by thread 12
PI 3.148000 calculated by thread 13
PI 3.156000 calculated by thread 14
PI 3.176000 calculated by thread 15
PI 3.146000 calculated by thread 16
PI 3.180000 calculated by thread 17
PI 3.126000 calculated by thread 18
PI 3.110000 calculated by thread 19
PI estimated on values: 3.141600
```

Rysunek 5: Wartości obliczone przez każdy z 20 wątków na podstawie 2000 punktów

```
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB6$ ./zad7
Created 20 threads.
PI 3.185333 calculated by thread 0
PI 3.149333 calculated by thread 1
PI 3.122667 calculated by thread 2
PI 3.153333 calculated by thread 3
PI 3.148000 calculated by thread 4
PI 3.129333 calculated by thread 5
PI 3.121333 calculated by thread 6
PI 3.153333 calculated by thread 7
PI 3.116000 calculated by thread 8
PI 3.153333 calculated by thread 9
PI 3.120000 calculated by thread 10
PI 3.145333 calculated by thread 11
PI 3.172000 calculated by thread 12
PI 3.114667 calculated by thread 13
PI 3.084000 calculated by thread 14
PI 3.098667 calculated by thread 15
PI 3.104000 calculated by thread 16
PI 3.082667 calculated by thread 17
PI 3.106667 calculated by thread 18
PI 3.168000 calculated by thread 19
PI estimated on values: 3.131400
```

Rysunek 6: Wartości obliczone przez każdy z 20 wątków na podstawie 3000 punktów