

Zajęcia PRIR nr 2

-- sprawozdanie

Jan Dobrowolski, 299242

Cel zajęć

W ramach drugich zajęć należało stworzyć program, który stworzy zadaną na wejściu liczbę procesów potomnych, które policzą sumę liczb zapisanych w pliku tekstowym. Do realizacji zadania należało użyć jednego mechanizmów IPC: pamięci współdzielonej.

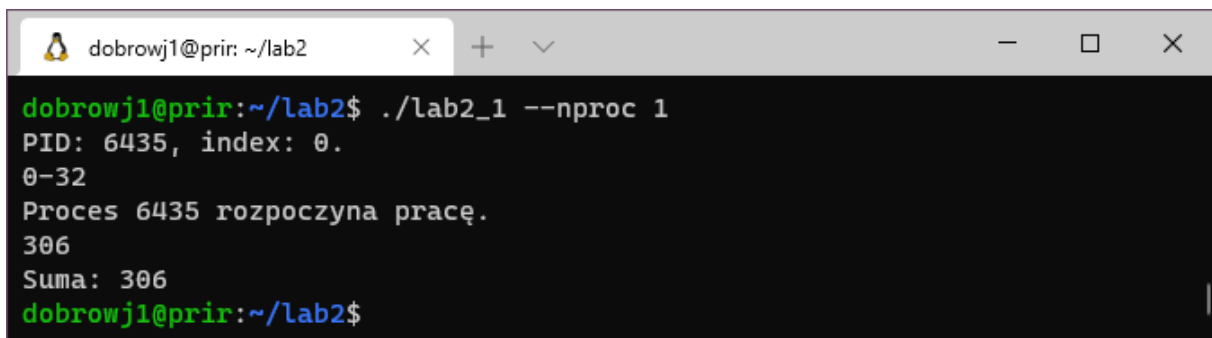
Zadanie 1

W pliku `zad1.c` umieściłem kod, który z pewnymi odstępstwami realizuje scenariusz działania zamieszczony w materiałach do ćwiczenia. Zamiast dwóch bloków pamięci współdzielonej wykorzystałem trzy: *range*, gdzie przechowywane były zakresy liczb do zsumowania przez podprocesy, *result*, gdzie przechowywane były obliczone sumy cząstkowe, oraz *init_data*, czyli wektor, do którego program przepisuje wartości z czytanego pliku (*vector.dat*). W rzeczonym pliku znajdują się 33 liczby o sumie równej 306.

Do określenia liczby tworzony procesów potomnych wykorzystać można flagę *nproc*. Jeśli się ją pominie, to program stworzy trzy takie procesy.

Oprócz danych podprocesów, program wypisuje podział zakres indeksów tablicy *init_data*, dla których korespondujące wartości są liczone przez dany proces.

Efekty wywołania programu:



```
dobrowj1@prir: ~/lab2
dobrowj1@prir:~/lab2$ ./lab2_1 --nproc 1
PID: 6435, index: 0.
0-32
Proces 6435 rozpoczyna pracę.
306
Suma: 306
dobrowj1@prir:~/lab2$
```

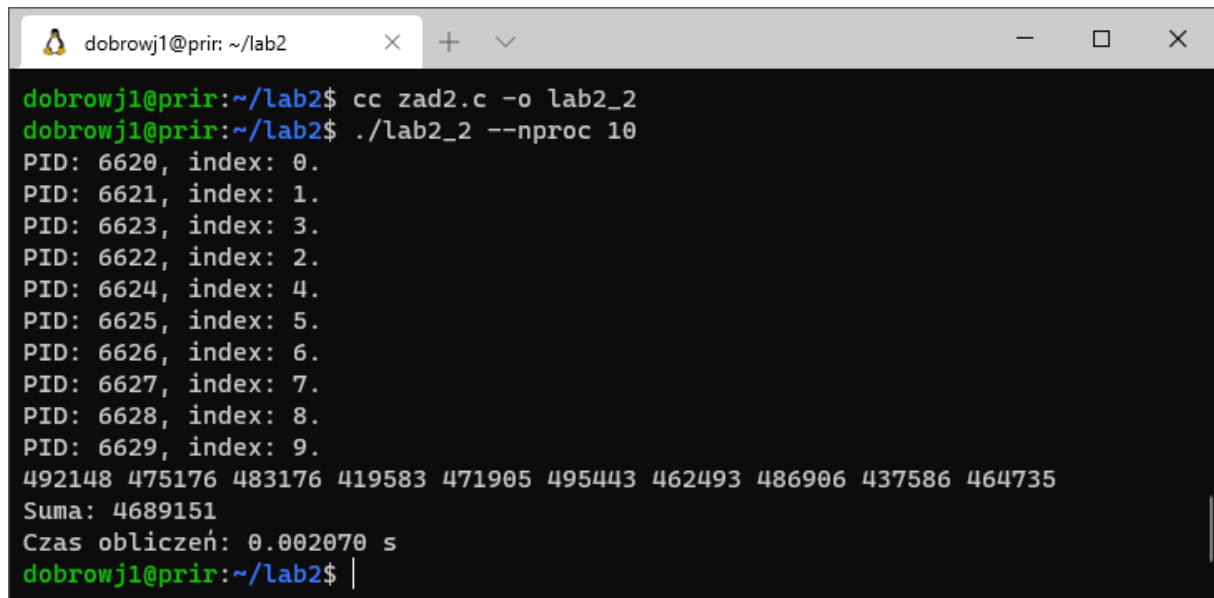
```
dobrowj1@prir: ~/lab2
dobrowj1@prir:~/lab2$ ./lab2_1 --nproc 3
PID: 6379, index: 0.
PID: 6380, index: 1.
PID: 6381, index: 2.
0-10 11-21 22-32
Proces 6379 rozpoczyna pracę.
Proces 6380 rozpoczyna pracę.
Proces 6381 rozpoczyna pracę.
110 110 86
Suma: 306
dobrowj1@prir:~/lab2$
```

```
dobrowj1@prir: ~/lab2
Suma: 306
dobrowj1@prir:~/lab2$ ./lab2_1 --nproc 10
PID: 6456, index: 0.
PID: 6457, index: 1.
PID: 6458, index: 2.
PID: 6459, index: 3.
PID: 6460, index: 4.
PID: 6461, index: 5.
PID: 6462, index: 6.
PID: 6463, index: 7.
PID: 6464, index: 8.
PID: 6465, index: 9.
0-5 6-8 9-11 12-14 15-17 18-20 21-23 24-26 27-29 30-32
Proces 6456 rozpoczyna pracę.
Proces 6457 rozpoczyna pracę.
Proces 6458 rozpoczyna pracę.
Proces 6459 rozpoczyna pracę.
Proces 6460 rozpoczyna pracę.
Proces 6461 rozpoczyna pracę.
Proces 6462 rozpoczyna pracę.
Proces 6463 rozpoczyna pracę.
Proces 6464 rozpoczyna pracę.
Proces 6465 rozpoczyna pracę.
60 30 30 30 30 30 30 30 30 6
Suma: 306
dobrowj1@prir:~/lab2$
```

Zadanie 2

W pliku `zad2.c` umieściłem kod z poprzedniego zadania zmodyfikowany tak, by liczył on również czas obliczeń wykonywanych przez procesy potomne z dokładnością do mikrosekund. Program odejmuje od uzyskanego pomiaru sześć sekund, czyli czas użycia funkcji `sleep()`.

Program wywołuje się w taki sam sposób, jak ten z poprzedniego zadania:



```
dobrowj1@prir: ~/lab2
dobrowj1@prir:~/lab2$ cc zad2.c -o lab2_2
dobrowj1@prir:~/lab2$ ./lab2_2 --nproc 10
PID: 6620, index: 0.
PID: 6621, index: 1.
PID: 6623, index: 3.
PID: 6622, index: 2.
PID: 6624, index: 4.
PID: 6625, index: 5.
PID: 6626, index: 6.
PID: 6627, index: 7.
PID: 6628, index: 8.
PID: 6629, index: 9.
492148 475176 483176 419583 471905 495443 462493 486906 437586 464735
Suma: 4689151
Czas obliczeń: 0.002070 s
dobrowj1@prir:~/lab2$
```

W odróżnieniu od programu `zad1.c`, tutaj czytany jest plik `vector2.dat`, który zawiera tysiąc liczb losowych, wygenerowanych poprzez stronę `random.org`.

Pomiary czasów dla każdej z liczb N [1, 2, 4, 6, 8, 16] powtórzyłem pięciokrotnie, po czym obliczyłem średnie czasy wykonania obliczeń. Uzyskane w ten sposób wyniki zebrałem w tabelce, która znajduje się na następnej stronie:

N	Próba	Wynik [μ s]	Średni czas dla N procesów [μ s]
1	1	373	403,8
	2	445	
	3	398	
	4	397	
	5	406	
2	1	482	534,6
	2	486	
	3	489	
	4	607	
	5	609	
4	1	1075	975,6
	2	931	
	3	880	
	4	851	
	5	1141	
6	1	1611	1335,6
	2	1313	
	3	1249	
	4	1287	
	5	1218	
8	1	1889	1816,6
	2	2092	
	3	1392	
	4	1864	
	5	1846	
16	1	3005	3315,6
	2	3560	
	3	3960	
	4	2854	
	5	3199	

Teoretycznie czas obliczeń powinien wynosić x/N , gdzie x – czas obliczeń dla $N = 1$, w rzeczywistości jednak czas obliczeń rośnie wraz ze zwiększeniem liczby procesów.

Jak widać na poniższym wykresie, czas obliczeń dla N rośnie liniowo:

