Zadanie 5

Modyfikacja mająca na celu spowodowanie, by wątki zwracały różne wartości zadziałała poprawnie, jednak wątki także przed modyfikacją zwracały różne wartości, zwracały swoje identyfikatory. Zaimplementowana modyfikacja spowodowała, że wątki zwracają rezultat wykonanych w pętli działań, który jest teraz zależny od identyfikatora, rzutowany na typ *long*.

Programy różnią się użytymi trybami wątków. W programie *join.c* użyto wątków z atrybutem **PTHREAD_CREATE_JOINABLE**, co powoduje, że proces główny czeka, aż wątki zwrócą mu przydzielone zasoby (odbywa się to za pomocą funkcji *pthread_join()*. W programie *detached.c* za to użyto wątków z atrybutem **PTHREAD_CREATE_DETACHED**, co powoduje, że wątki automatycznie zwracają zasoby po zakończeniu.

Zadanie 6

Po kompilajcji i próbie uruchomienia programu występuje naruszenie ochrony pamięci. Dzieje się tak, ponieważ każdy wątek tworzy tablicę typu *double* z dwoma milionami elementów. Jako że typ *double* zajmuje 8 bajtów, to taka tablica zajmowała by powyżej 15 megabajtów. Błąd występuje, ponieważ maksymalny rozmiar stosu jest równy 8192 kilobajty (8 megabajtów).

Aby rozwiązać ten problem, możemy zmiejszyć rozmiar tablicy tak, aby nie powodował przekroczenia dostępnej pamięci (np. zmniejszyć rozmiar o połowę). Nie jest to dobre rozwiązanie, jeśli nie możemy sobie pozwilić na zmniejszenie rozmiaru tablicy np. przez założenia programu.

Innym rozwiązaniem jest zwiększenie limitu pamięci stosu za pomocą polecenia *ulimit -s 16384*. Drugim argumentem jest rozmiar w kilobajtach, na jaki chcemy zmienić maksymalny rozmiar stosu. Jest to rozwiązanie uniwersalne, ograniczone jedynie przez możliwości systemu komputerowego, na którym pracujemy i nie wymaga od nas ingerencji w kod programu, jednak wymaga od nas posiadania konkretnych uprawnień.

Najlepszym rozwiązaniem jest modyfikacja programu w taki sposób, aby tworzone wątki miały powiększony rozmiar stosu. Możemy to uzyskać poprzez użycie funkcji *pthread_attr_setstacksize()*, gdzie jako pierwszy argument podajemy referencję do struktury zaierającej atrybuty wątków, a jako drugi podajemy pożądany rozmiar stosu. W tym przypadku może to być wartość większa od liczbiy elementów tablicy, pomnożonych przez rozmiar zmiennej *double* (osiem) o np. 1 megabajt. Ten dodatkowy megabajt przeznaczony jest na inne dane, które mogą znaleźć się na stosie podczas wykonywania programu, podczas gdy 16 megabajtów będzie używane przez tablicę. Po takim zaimplementowaniu funkcji należy pamiętać, aby przy tworzeniu wątków funkcją *pthread_create()* podać jako drugi argument referencję do struktury zawierającej atrybuty wątków, w której ustawiliśmy powiększony rozmiar stosu.