Magdalena Zając

RAPORT

Zarządzanie pamięcią, biblioteki, pomiar czasu

Zadanie 1. Kompilator, optymalizacja, pomiar czasu

Program 1: fibo.c

Ciąg Fibbonacciego

Flaga	Czas wykonania (s)
	35,780594
O	3,068055
O1	3,259160
O2	3,544509
O3	3,362105

Możemy zaobserwować tutaj skrócenie czasu wykonania zoptymalizowanego programu niemal dziesięciokrotnie w porównaniu do programu który został skompilowany bez żadnych flag.

Program 2: rek.c

Rekurencja ogonowa

Flaga	Czas wykonania (s)
	21,661612
O	9,134144
O1	9,142607
O2	0.000000
O3	0.000000

W tym przypadku optymalizacja O2 i O3 spowodowały ogromny wzrost szybkości działania programu.

Program 3: for.c

Pętla FOR

Flaga	Czas wykonania (s)
	2,065286
O	1,774040
O1	1,780880
O2	1,774617
O3	1,856272

Program wykonuje jedynie iterację. Jego zoptymalizowana postać wykonuje się nieznacznie szybciej, jednak w przypadku pisania większych programów z wieloma iteracjami, ta różnica może okazać się istotna.

Program 4: bubble.c

Bubble sort

Flaga	Czas wykonania (s)
	0,242719
O	0,148011
O1	0,141169
O2	0,142202
O3	0,140998

Program, tak jak poprzednie zyskał na optymalizacji. Jednak różnice pomiędzy flagami są niewielkie.

Podsumowanie:

Optymalizacja pozwala zwiększenie wydajności programu, jednak wyższe stopnie optymalizacji wydłużają czas kompilacji. Wybór optymalizacji powinien więc być przemyślany pod kątem tego, co tą optymalizacją chcemy uzyskać.

Zadanie 2. Program korzystający z biblioteki