METODY PROGRAMOWANIA – LABORATORIUM 6

PROGRAM 01

Dane: **n>1** oraz tablica liczb całkowitych: **A[0], ... , A[n-1]**.

Inwersją w tablicy **A** nazywamy każdą parę liczb (i, j): i < j dla których A[i] > A[j].

Korzystając z metody **„dziel i zwyciężaj”** napisz program zawierający **rekurencyjną** funkcję działającą w czasie **O(n log n)**, która wyznacza liczbę inwersji w tablicy **A**.

WEJŚCIE

Plik testowy o dowolnej nazwie (standardowe wejście) ma następujący format:

- Pierwsza linia zawiera dodatnią liczba całkowitą oznaczająca ilość zestawów danych, po której na wejściu pojawią się zestawy danych w ilości równej wczytanej liczbie.
- Każdy zestaw danych zawiera:
  - dodatnią liczbę całkowitą z zakresu od 1 do 215 oznaczającą ilość danych wczytywanego zestawu,
  - zasadnicze dane zestawu w ilości równej poprzednio wczytanej wartości, będące liczbami całkowitymi z zakresu od -2^{48} do $+2^{48}$.

WYJŚCIE

Dla każdego zestawu danych wypisz ilość inwersji w zadanej tablicy.

Uwagi: Podziel program na funkcje. Pamiętaj o komentarzach do kodu

PRZYKŁADY

Wejście:

```
4
10
1 2 3 4 5 6 7 8 9 10
10
10 1 2 3 4 5 6 7 8 9
10
10 9 8 7 6 5 4 3 2 1
10
0 0 0 0 0 0 0 0 0 0
```

Wyjście:

```
0
9
45
0
```


PROGRAM 02

Napisz program generujący liczby losowe za pomocą podanych poniżej metod.

Program powinien umożliwiać:

- wybór jednej z metod: Liniowa metoda kongruencyjna, Addytywna metoda kongruencyjna, Metoda tasowania z jedną sekwencją losową
- wygenerowanie ciągu n liczb pseudolosowych
- dobór ziarna (seed) w postaci liczby całkowitej dla każdej z metod
- zapisanie wygenerowanego ciągu liczb do pliku tekstowego w postaci kolumny

Liniowa metoda kongruencyjna

$$X_{n+1} = (a \times X_n + c) \bmod m$$

X_{n+1} – n-ta liczba pseudolosowa
X_n – poprzednia liczba pseudolosowa (wynik poprzednich obliczeń będący jednocześnie stanem wewnętrzny generatora)
a – mnożnik (odpowiednio dobrany)
c – przyrost (odpowiednio dobrany , gdy c!=0 mówimy, że to LCG addytywny, gdy c==0, że jest to LCG multiplikatywny)
m – moduł (zakres liczb generowanych)

Wartość początkowa x0 (seed)) jest dowolna liczba całkowita z zakresu obejmującego okres generatora.

Uwaga!

- W przypadku generatora LCG należy napisać program wyliczający współczynniki tego generatora: m, a, c na podstawie maksymalnej liczby pseudolosowej Xmax, którą należy podać na wejściu.

PRZYKŁAD LABORATORIUM

1. Zaprojektować generator bazujący na liniowej metodzie kongruencyjnej (LCG) generujący liczby pseudolosowe w dowolnym przedziale.
2. Zaprojektować generator bazujący na addytywnej metodzie kongruencyjnej generujący liczby pseudolosowe w dowolnym przedziale.
3. Zaprojektować generator bazujący na metodzie tasowania z jedną sekwencją losową generujący liczby pseudolosowe w dowolnym przedziale.

Informatyka, rok I, sem. II - 2017/2018 – Katedra Automatyki i Technik Informacyjnych (E-3) – Politechnika Krakowska
mgr inż. Grzegorz Nowakowski

ZASADY ODDAWANIA GOTOWYCH PROGRAMÓW:

Plik ***.cpp*** o nazwie:

Nazwisko_Imie_Program_01.cpp
Nazwisko_Imie_Program_02.cpp

wraz z wszystkimi plikami oraz plikami tekstowymi powinny być zamieszczone w katalogu: ***Nazwisko_Imie_Laboratorium_6***

Katalog powinien być spakowany w formacie ***.rar*** lub ***.zip*** i przesłany do folderu: ***Programy - laboratorium 6 – Poniedziałek godzina 15.30*** dostępnego na stronie kursu MP (elf2.pk.edu.pl).

LITERATURA:

Wieczorkowski R., Zieliński R. : Komputerowe generatory liczb losowych, WN-T, Warszawa 1997

Knuth Donald E. : Sztuka programowania. T. 2, Algorytmy seminumeryczne, WN-T, Warszawa 2002

DODATEK:

Algorytmy "dziel i zwyciężaj"

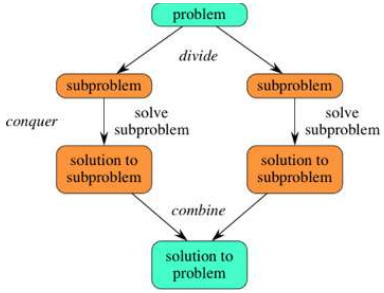
Zasada działania algorytmów tego typu polega na podzieleniu problemu na mniejsze, a następnie niezależnego rozwiązania każdego z nich. Rozwiązanie każdego z mniejszych problemów jest takie samo – jest on dzielony na jeszcze mniejsze problemy, które następnie zostają rozwiązane niezależnie od siebie.

Ten proces dzielenia na coraz to mniejsze problemy jest kontynuowany aż do momentu uzyskania przypadku bazowego, który można rozwiązać w prosty sposób, bez dalszego dzielenia zagadnienia.

Tak otrzymane rozwiązania poszczególnych podzadań są następnie scalane tworząc rozwiązanie całego zadania.

Metoda „dziel i zwyciężaj" jest bardzo często używana w algorytmach rekurencyjnych. Metody działające zgodnie z tą zasadą zawierają zazwyczaj dwa rekurencyjne wywołania samej siebie, z których każde obsługuje „połowę" problemu.

Przykład: Jeden krok algorytmu, przy założeniu, że operacja dzielenia problemu tworzy dwa podproblemy:



Generacja liczb pseudolosowych

```
#include<iostream>
#include <random>
using namespace std;

int main() {
	random_device rd; //uruchomienie generator liczb pseudolosowych
	mt19937 mt(rd()); //o nazwie Mersenne Twister


	//generacja liczb całkowitych
	uniform_int_distribution<int>dist(1, 10);  //zawężenie generowania liczb
                                                   //do przedziału (1,10)


	//generacja liczb rzeczywistych
	//uniform_real_distribution<double> dist(1, 10);


	for (int i = 0; i<16; ++i)
		cout << dist(rd) << "\t" << dist(mt) << "\n";


	cin.ignore();
	getchar();
	return 0;
}
```

Powyższe linijki kodu są odpowiedzialne za uruchomienie generator liczb pseudolosowych o nazwie Mersenne Twister – w skrócie – MT. Generator MT jest jednym, dużym rejestrem przesuwnym ze sprzężeniami zwrotnymi powodującymi rotację oraz mieszanie się bitów. Rejestr ma długość 19937 bitów (liczba Mersenne'a), które w pamięci zajmują 624 słowa 32-bitowe.