

Metody Monte Carlo - laboratorium 1

Zadanie 1

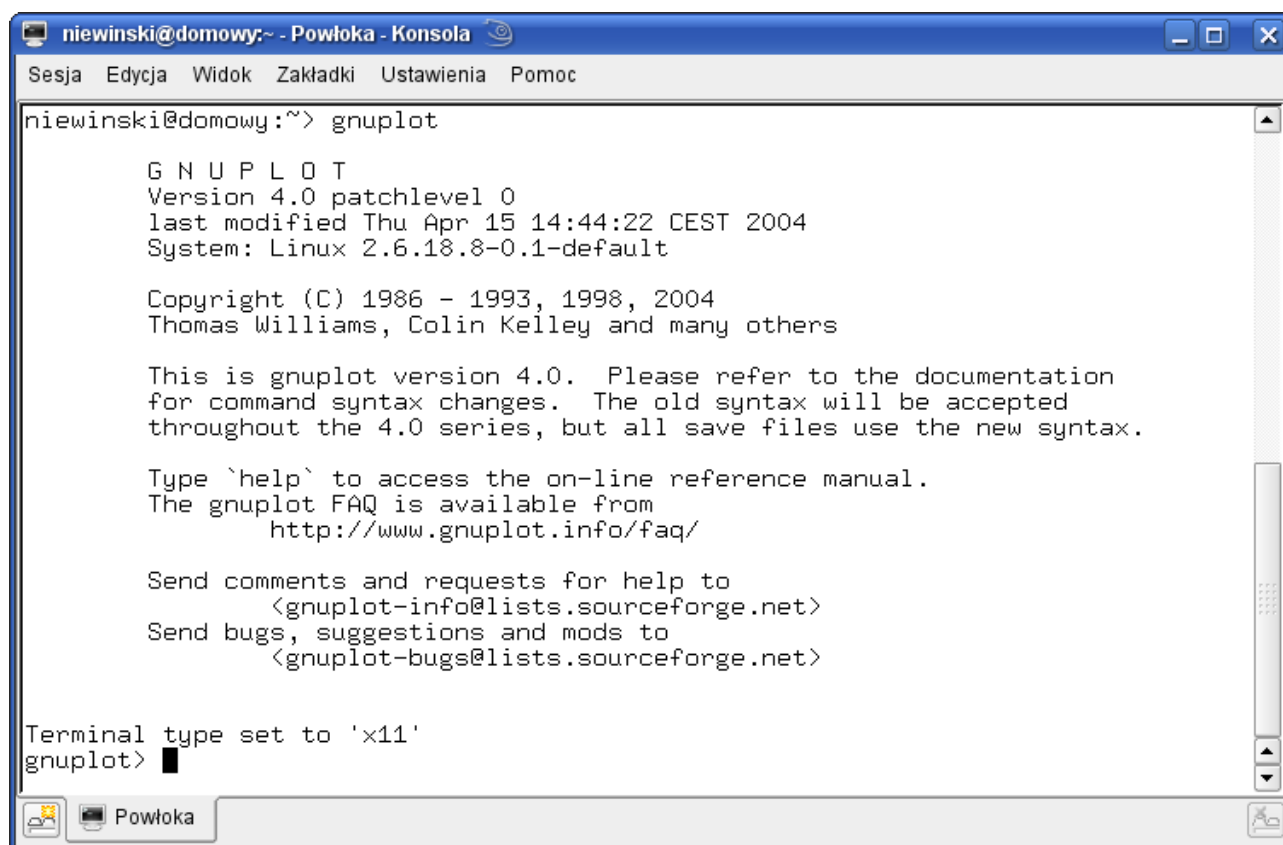
Napisać implementację generatora multiplikatywnego w oparciu o zależność:

$$X_n = (a_1 X_{n-1} + c) \bmod m$$

gdzie: $m = 2^{13} - 1$, $a_1 = 17$ i $c = 0$.

Następnie proszę wygenerować 2500 liczb i narysować strukturę przestrzenną generatora (w postaci par: $(x, y) = (x_{i+1}, x_i)$).

Uwaga: w tym celu proszę programowo stworzyć plik tekstowy (np. 1.txt), który w każdej linii będzie zawierał współrzędne jednego punktu (x, y) przy czym wartości x i y mają być rozdzielone znakiem spacji. Następnie w środowisku **gnuplot** wywołać polecenie `plot "1.txt"`. (środowisko to uruchamiamy w terminalu wywołując polecenie **gnuplot** – patrz rysunek)



```
niewinski@domowy:~> gnuplot

G N U P L O T
Version 4.0 patchlevel 0
last modified Thu Apr 15 14:44:22 CEST 2004
System: Linux 2.6.18.8-0.1-default

Copyright (C) 1986 - 1993, 1998, 2004
Thomas Williams, Colin Kelley and many others

This is gnuplot version 4.0. Please refer to the documentation
for command syntax changes. The old syntax will be accepted
throughout the 4.0 series, but all save files use the new syntax.

Type `help` to access the on-line reference manual.
The gnuplot FAQ is available from
  http://www.gnuplot.info/faq/

Send comments and requests for help to
  <gnuplot-info@lists.sourceforge.net>
Send bugs, suggestions and mods to
  <gnuplot-bugs@lists.sourceforge.net>

Terminal type set to 'x11'
gnuplot>
```

Wykonać to samo dla parametru $a_1 = 85$. Następnie proszę narysować strukturę przestrzenną generatora (w postaci par: $(x, y) = (x_{i+2}, x_i)$). Jako x_0 przyjąć np. 19.

Metody Monte Carlo - laboratorium 1

Zadanie 2

Napisać implementację generatora złożonego w oparciu o algorytm Wichmanna and Hilla

$$X_n = 171 X_{n-1} \bmod 30269,$$

$$Y_n = 172 Y_{n-1} \bmod 30307,$$

$$Z_n = 170 Z_{n-1} \bmod 30323,$$

$$K_n = \left(\frac{X_i}{30269} + \frac{Y_i}{30307} + \frac{Z_i}{30323} \right) \bmod 1, \text{ okres } K \sim 10^{12}$$

i narysować strukturę przestrzenną generatora dla par (x_{i+1}, x_i) i (x_{i+2}, x_i) (na podstawie wylosowanych 500 liczb pseudolosowych).

Zadanie 3

Skompilować i uruchomić poniższy przykład uzupełniając go o wywołanie funkcji:

`gsl_rng_set (const gsl_rng * r, unsigned long int s)`

ustawiającej wartość “zarodka” generatora (np. 123). W miejsce wstawić nazwę wybranego algorytmu dostępnego w bibliotece GSL.

```
#include <iostream>
#include <gsl/gsl_rng.h>
using namespace std;
int main(void) {
    gsl_rng * generatorek;
    generatorek = gsl_rng_alloc(.....);
    for (int i = 0; i < 100; i++)
        cout << gsl_rng_uniform(generatorek) << endl;
    gsl_rng_free(generatorek);
    return 0;
}
```

Uwaga: listę dostępnych typów generatorów można uzyskać przy użyciu funkcji

`const gsl_rng_type **gsl_rng_types_setup(void)`. Przykład jej użycia:

```
const gsl_rng_type **t, **t0;
t0 = gsl_rng_types_setup ();
```

Metody Monte Carlo - laboratorium 1

```
printf ("Dostępne typy generatorów:\n");  
for (t = t0; *t != 0; t++) printf ("%s\n", (*t)->name);
```

Uwaga: przy kompilacji należy dołączyć wymagane biblioteki: (-lgsl -lgslcblas -lm)

Zadanie 4

Bazując na doświadczeniach uzyskanych w zad. 3 wyznaczyć względne czasy generacji liczb (np. 10^6 - 10^9 – w zależności od mocy komputera) poszczególnych algorytmów biblioteki GSL, odnosząc je do czasu uzyskanego przy użyciu generatora typu **gsl_rng_default**. Do pomiaru czasu proszę wykorzystać funkcję **gettimeofday()**. Wypełnia ona strukturę typu **timeval** przekazaną jej jako argument. Struktura ta ma dwa pola: **tv_sec** (liczba sekund, które upłynęły od początku 1970 roku) oraz **tv_usec** (liczba mikrosekund, które upłynęły od zakończenia ostatniej pełnej sekundy). Przykład jej użycia do wyznaczenia czasu trwania operacji:

```
#include <sys/time.h>  
timeval tstart, tstop;  
gettimeofday(&tstart, 0);  
// operacje których czas wykonywania chcemy zmierzyc  
gettimeofday(&tstop, 0);  
time_diff_millis(tstart, tstop);  
  
long time_diff_millis(timeval &tstart, timeval &tstop) {  
    return (long)(  
        (1000 * tstop.tv_sec - 1000 * tstart.tv_sec)  
        + (tstop.tv_usec - tstart.tv_usec) / 1000  
    );  
}
```

Prędkości proszę przedstawić w postaci wykresu słupkowego wygenerowanego przez program **gnuplot**. W tym celu proszę stworzyć programowo plik tekstowy zawierający w liniach:
numer_generatora względny_czas_generacji # nazwa_generatora
i wyświetlić go programem *gnuplot* za pomocą polecenia
plot "nazwa_pliku" with boxes