Języki programowania, 9.12.2020

Tym razem napiszemy coś bardziej praktycznego – klasę do generowania liczb pseudolosowych z dwóch (a jak starczy czasu to z trzech) różnych rozkładów - jednorodnego, Gaussa oraz Poissona.

Wirtualność

Wirtualność oraz polimorfizm (tu rozumiany jako możliwość wyboru postaci funkcji w trakcie działania programu) stanowią zupełnie inne podejście do programowania niż programowanie proceduralne, które do tej pory wykorzystywaliśmy. Są najważniejszą cechą programowania orientowanego obiektowo (OOP – Object Oriented Programming, inaczej mówiąc: **orientującego się według typu obiektu**) takiego jak C++ i mają ogromne możliwości, które objawiają się w pełni przy dużych projektach.

Tworzymy klasę abstrakcyjną (ATD – abstrakcyjny typ danych) – 1 p.

Tworzymy klasę abstrakcyjną `Random`, która zawiera konstruktor domyślny, pusty destruktor, oraz metodę metodę czysto wirtualną `double Exec()`, tzn. w pliku Random.h:

```
virtual double Exec() = 0;
```

W konstruktorze ustawiamy ziarno losowania `srand(time(NULL))`.

Wskazówka: wszystkie metody czysto wirtualne muszą istnieć w klasach pochodnych.

Klasa, która ma co najmniej jedną funkcję czysto wirtualną nazywamy **klasą abstrakcyjną**.

Po co tworzyć klasy abstrakcyjne?

Czasem mamy jakiś obiekt, który łączy cechy kilku innych obiektów (jak np. nasza klasa `Random`) ale sam w sobie nie jest kompletnym obiektem. W naszym przypadku mamy więc klasę `Random`, która zawiera wspólne elementy dla klas podrzędnych (tj. ustawianie ziarna). Do samego losowania mamy zaś różne inne klasy podżędne służące do generowania konkretnych liczb w konkretny sposób: jednorodnie, z rozkładu Gaussa, Poissona, itd.

Klasa abstrakcyjna jest klasą jakby niedokończoną. Jej dokończenie realizowane jest przez klasy pochodne.

Klasa do generowania liczb losowych z rozkładu jednorodnego – 1 p.

Klasa będzie się nazywać `Uniform` i dziedziczy z klasy `Random`. Jej plik nagłówkowy powinien zawierać:

- pola `double fMin` i `double fMax` – przedział, z którego będą losowane liczby
- konstruktor domyślny – ustawia ziarno z zegara systemowego poprzez użycie konstruktora klasy nadrzędnej oraz ustawia `fMin` na 0 i `fMax` na 1
- Konstruktor z parametrami ustawiający pola `fMin` i `fMax`; `srand` ustawiamy analogicznie jak w konstruktorze bez parametrów.
- metody „set" i „get" ustawiające pola `fMin` i `fMax`
- metoda `double Exec()` - metoda wirtualna, zwraca liczbę pseudolosową z rozkładu jednorodnego z przedziału <fMin;fMax)

Klasa do generowania liczb losowych z rozkładu Gaussa – 2 p.

Następnie tworzymy klasę `Gauss`, która dziedziczy z klasy `Random`. Powinna ona zawierać:

- pola typu `double fMean`, `double fSigma`, które przechowują parametry rozkładu Gaussa: średnią i odchylenie standardowe.
- konstruktor z parametrami ustawiający wszystkie pola jak
- metody typu „set" ustawiające oba pola i metody „get" zwracające oba pola
- metodę `double Exec()` - zwracającą liczbę pseudolosową z rozkładu Gaussa o zadanej średniej i odchyleniu standardowym (szczegóły implementacji patrz **Uwaga1!**)

Jeśli starczy czasu po zrobieniu pozostałych poleceń, dokładamy trzecią klasę `Poisson`, losującą liczby pseudolosowe z rozkładu Poissona (patrz **Uwaga 2!**).

Zestaw bibliotek niezbędnych do stworzenia generatora:

```
cstdlib, ctime, cmath
```

Należy użyć funkcji `rand()` - zwracającej liczbę całkowitą pseudolosową z rozkładu jednorodnego <0;RAND_MAX), gdzie `RAND_MAX` jest wielką liczbą - stałą zdefiniowaną w bibliotece standardowej.

Piszemy program testujący działanie dwóch wyżej wymienionych klas – 1 p.

Tworzymy w funkcji `main` obiekt klasy `Uniform` (np. `Uniform uni1(-2,3);`) z ustawionymi wartościami od -2 do 3. Losujemy kilka liczb pseudolosowych i wypisujemy na ekran.

Tworzymy w funkcji `main` obiekt klasy `Gauss` (np. `Gauss gaus1(100,5);`) z ustawioną średnią na 100 i odchyleniem 5. Losujemy kilka liczb pseudolosowych.

Tworzymy wskaźnik na obiekt klasy `Random` i ustawiamy go na wcześniej stworzony obiekt tej klasy:

```
Random *rand = &uni1;
```

i wypisujemy kilka losowych wartości metodą `Exec()`. Następnie ten sam wskaźnik `rand2` ustawiamy na obiekt klasy `Gauss`:

```
rand = &gaus1;
```

Obsługa plików tekstowych – 1 p.

W programie zapisujemy kilka wylosowanych liczb z rozkładu Gaussa do pliku tekstowego. Następnie otwieramy zapisany plik i wczytujemy liczby oraz wypisujemy je na ekran.

Przykład zapisywania do pliku:

```
#include <fstream> //naglowek zawierajacy funkcje C++ do operacji na plikach

ofstream ofile;
ofile.open("file.txt"); //"file.txt" jest tutaj typu char*
ofile<<"tekst zapisany do pliku"<<endl;
ofile.close();
```

Przykład wczytywania liczb z pliki:

```
#include <fstream>

ifstream ifile;
int val;
ifile.open("file.txt");
while(ifile>>val)
{
      cout<<"val: "<<val<<endl;
}
ifile.close();
```

Uwaga 1!

Algorytm do generowania liczb pseudolosowych z rozkładu Gaussa (metoda Box-Muller'a):

- U, V – dane liczby pseudolosowe z rozkładu jednorodnego <0;1)
- Liczba losowa z rozkładu Gausa o średniej μ i odchyleniu standardowym σ dana jest wtedy wzorem:

$$x = \mu + \sigma\sqrt{-2\ln(U)}\cos(2\pi V)$$

Uwaga 2!

Algorytm do generowania liczb pseudolosowych z rozkładu Poissona o średniej λ :

- $x_0, x_1, \ldots$ - ciąg liczb pseudolosowych z rozkładu jednorodnego
- Jeżeli $x_0 x_1 \ldots x_k < e^{-\lambda}$, to k jest liczbą z rozkładu Poissona.