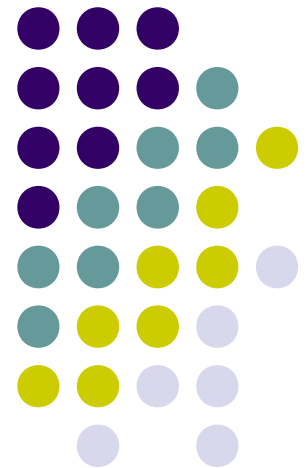


Średniozaawansowane programowanie w C++

Program #2
20 października 2016 r.
na
3 listopada 2016 r.



Zadanie



Napisać program wczytujący listę plików z liczbami (podawany jako pierwszy parametr programu) i obliczający dla nich estymatory rozkładów Gaussa, Lorentza i Poissona.

Użytkownik powinien najpierw wybrać z listy plik z danymi do analizy, a następnie rodzaj rozkładu, dla którego ma policzyć statystykę.

Podział na jednostki kompilacji



1. main - dosyć długi

- Obsługa parametrów startowych programu
- Wczytanie listy plików
- Rejestracja dostępnych klas liczących estymatory
- Wyświetlanie i obsługa menu
- Wyświetlanie obliczeń

2. dane_stat

- Trzy klasy (bazowa, proxy, „prawdziwa”)
- wczytujące i przechowujące dane z plików

3. rozklad

- class Rozklad – klasa bazowa dla obliczaczy estymatorów
- class Rozklad* – klasy pochodne reprezentujące konkretne rozkłady
- class FabrykaRozkladow – nic dodać, nic ująć :)

main.cpp – fragmenty



```
// Wskaźniki do obiektów przechowujących dane  
std::vector <boost::shared_ptr <DaneStat> > dane;
```

WAŻNA SPACJA!

```
// Rejestrujemy wtyczki  
FabrykaRozkladow::rejestruj (&RozkladGaussa::kreator, std::string  
("Rozklad Gaussa"));  
FabrykaRozkladow::rejestruj (&RozkladLorentza::kreator, std::string  
("Rozklad Lorentza"));  
FabrykaRozkladow::rejestruj (&RozkladPoisson::kreator, std::string  
("Rozklad Poissona"));
```

```
// Tworzy miziadelko do obliczania statystyk  
std::auto_ptr <Rozklad> rozkl (FabrykaRozkladow::utworz (wybor_r, dane  
[wybor-1]->dane ()));
```

dane_stat.hpp



```
#ifndef _dane_stat_hpp_
#define _dane_stat_hpp_

#include <string>
#include <vector>

class DaneStat          // klasa bazowa
{
    public:
        DaneStat (const std::string &nazwa);
        // zwraca referencje na przechowywane dane
        virtual const std::vector <float> &dane () const = 0;
        virtual ~DaneStat () {};
        virtual const std::string &nazwa () const;           // zwraca nazwe pliku
    protected:
        std::string nazwa_; // nazwa pliku
};

// proxy - wczytuje prawdziwy obiekt przy pierwszym użyciu
class DaneStatProxy : public DaneStat
{
};

// "prawdziwy" obiekt przechowujący dane
class DaneStatReal : public DaneStat
{
};

#endif
```

rozkład.hpp



```
// Mapa przechowująca obliczone estymatory rozkładu: opis-wartosc
typedef std::map <std::string, float> ParametryRozkładu;

class Rozkład          // ATD - klasa bazowa dla "obliczaczy" estymatorów
{
    public:
        // ustawia referencje na dane do analizy
        explicit Rozkład (const std::vector <float> &dane);
        virtual ~Rozkład () {}
        // oblicza estymatory i zwraca je w mapie
        virtual std::auto_ptr <ParametryRozkładu> oblicz () const = 0;
    protected:
        // przechowuje referencje na dane do analizy
        const std::vector <float> &dane_;
};

class RozkładGaussa : public Rozkład
{
    public:
        // nie robi nic sensownego poza wywołaniem konstr. klasy bazowej z odpowiednim parametrem
        explicit RozkładGaussa (const std::vector <float> &dane);
        virtual ~RozkładGaussa () {}
        // liczy wartosc srednia i odchylenie standardowe
        virtual std::auto_ptr <ParametryRozkładu> oblicz () const;
        // statyczna met. tworząca i zwracająca wskaznik na obiekt własnego typu
        static Rozkład* kreator (const std::vector <float> &dane);
};
```

rozkład.hpp (2)



```
//      wskaznik typu KreatorRozkladu do funkcji tworzacej obiekt
//      obliczacza rozkladu (pochodna klasy Rozklad)
typedef Rozklad* (*KreatorRozkladu)(const std::vector <float> &);

class FabrykaRozkladow          // FABRYKA! :)
{
    private:
        // przechowuje wskaźniki kreatorow (funkcji tworzących!)
        static std::map <unsigned, KreatorRozkladu> rozklady;
        // przechowuje nazwy rozkladow
        static std::map <unsigned, std::string> nazwy;
    public:
        // rejestruje kreator danego rozkladu (id generowane przyrostowo od 1)
        static void rejestruj (KreatorRozkladu kr, const std::string &nazwa);
        // wola kreator dla rozkladu o wybranym id
        static Rozklad *utworz (unsigned id, const std::vector <float> &dane);
        // zwraca nazwe rozkladu o identyfikatorze id
        static std::string nazwa (unsigned id);
        // zwraca liczbe zarejestrowanych rozkladow
        static unsigned ilosc () {return rozklady.size ();}
};
```

rozkład.cpp



```
RozkładGaussa::RozkładGaussa (const std::vector <float> &dane) : Rozkład (dane) {}
```

```
// Statyczny kreator wybranej klasy (to naprawdę jest takie proste!)
```

```
Rozkład *RozkładGaussa::kreator (const std::vector <float> &dane)
```

```
{
```

```
    return new RozkładGaussa (dane);
```

```
}
```

```
// Statyczna metoda wołająca (choć trudno to zobaczyć) kreator typu o podanym id
```

```
Rozkład *FabrykaRozkładow::utworz (unsigned typ, const std::vector <float> &dane)
```

```
{
```

```
    return rozkłady [typ] (dane);
```

```
}
```


Programowanie jest fantastyczne!!!

