

Dzisiaj wykorzystamy kolejną bardzo ważną rzecz języka C++ - szablony klas. W tym celu wykorzystamy napisaną wcześniej klasę `List` tworzącą listę. Wtedy nasza klasa mogła przyjmować i przechowywać jedynie elementy typu `double`. Teraz chcemy rozszerzyć jej działanie na dowolne elementy.

Zasada działania naszej klasy z punktu widzenia programu będzie prawie identyczna jak listy ze standardowej biblioteki szablonów STL, np.:

```
List<double> *myList = new List<double>(); //tworzy liste
przechowujaca typ double za pomoca naszej klasy List
list<double> *myListSTL = new list<double>(); //tworzy liste
przechowujaca typ double za pomoca klasy z biblioteki STL
```

W podobny sposób chcemy przerobić klasę `Histogram`. Klasę tą napisaliśmy tak, by mogła przechowywać jedynie dane typu `int` (później pisaliśmy dwie klasy `HistogramDouble` i `HistogramInt` do przechowywania dwóch typów danych). Teraz natomiast, zamiast pisać dwie klasy, chcemy nasz histogram (jedną klasę) uogólnić, by mógł przyjmować elementy dowolnego typu. Jak to zrobić? Musimy go przerobić właśnie na klasę szablonoową. Na końcu chcemy również stworzyć abstrakcyjną klasę bazową `THistogram` dla klasy szablonoowej `Histogram`.

Do zrobienia:

- przerobienie klasy `List` tak, by stała się klasą szablonoową. Stworzenie w programie listy przy użyciu naszej klasy oraz listy z biblioteki STL na zmiennych typu `double`, dodanie do list 3 elementów, wypisanie obu list.
- Zmodyfikowanie napisanej poprzednio klasy `Histogram`. Tworzymy dwie listy (używając nasze klasy) przechowujące wskaźniki do histogramów (jedna do typu `Histogram<int>`, druga do typu `Histogram<double>`). Następnie tworzymy dwa histogramy, jeden `Histogram<double>` i drugi typu `Histogram<int>`, dodajemy je do list, które potem wypisujemy na ekran. Histogramy (mają być wskaźnikami!) należy stworzyć używając pliku z danymi wygenerowanymi przez klasę `Random` na poprzednich zajęciach. Nazwę tego pliku podajemy jako parametr uruchomienia programu.
- Trzeba zauważyć, że szablonoowa metoda `PrintList()` w klasie `List` nie zadziała w przypadku listy wskaźników na histogramy (wypisze na ekran adres komórki pamięci danego wskaźnika). Chcemy stworzyć więc specjalizowaną metodę, która będzie wypisywać poprawnie histogram na ekran używając metody `PrintHistogram()`. Tworzymy również globalną funkcję szablonoową `void CompareList()`, którą będzie porównywał rozmiar (używając metody `Capacity()`) dwóch list (mogą być to listy różnych typów, np. przechowującej `double` i przechowującej `int`) i wypisywać większy rozmiar na ekran. Używamy jej do porównania rozmiarów stworzonych list.
- Jeśli starczy czasu – napisać klasę abstrakcyjną `THistogram`, z której będzie dziedziczyć klasa szablonoowa `Histogram`. Pozwoli to na stworzenie listy, do której będzie można wstawić histogramy dowolnego typu (idealny przykład idei abstrakcyjnych typów danych i wirtualności).

Uwaga!

Należy posiadać poprawnie działającą wersję klasy `List` oraz poprawnie działającą klasę z histogramem – `Histogram`.

Klasy szablonoowe piszemy **tylko** w plikach `.h` – nie tworzymy pliku `.cpp` (ponieważ klasa szablonoowa jest kompilowana dopiero przy jej użyciu w trakcie wykonywania programu).

Używamy pliku `Makefile` w najbardziej zaawansowanej postaci.