Laboratorium JP

Projekt 4

Dziedziczenie, abstrakcja danych, hermetyzacja, szablony, wzorce

1. Wykorzystując klasę główną z projektu 3 zaprojektować i zrealizować zestaw klas składający się z **jednej** abstrakcyjnej klasy bazowej i **dwóch** klas pochodnych. Zaprojektować i zdefiniować **metody wirtualne** dla klasy bazowej i pochodnych.

Można wykorzystać swoją klasę główną jako abstrakcyjną klasę bazową lub potraktować ją jako jedną z klas dziedziczących.

Przykłady:

a) Klasą główną w projekcie 3 był „Samochód". Można ją przerobić na abstrakcyjną klasę bazową, po której będą dziedziczyć utworzone klasy „Ciężarówka" oraz „Kabriolet"

b) Klasą główną w projekcie 3 był „Ekspres do kawy". Można stworzyć abstrakcyjną klasę bazową „Urządzenie do produkcji napoju", po której „Ekspres do kawy" dziedziczy. Dodatkowo należy utworzyć drugą klasę pochodną, na przykład „Automat z puszkami"

2. Rygorystycznie potraktować deklaratory zakresu: private, protected i public, stosując zasadę najmniejszych przywilejów.
3. Należy napisać klasę Kontener, która będzie służyła do przechowywania obiektów dowolnego (ale z góry nie określonego) typu.
4. Należy napisać program pokazujący prawidłowe działanie dziedziczenia oraz wykorzystanie stworzonego kontenera do przechowywania obiektów utworzonych klas z punktu 1.

Założenia:

- kontener zrealizowany jest jako lista dwukierunkowa,
- kontener zaprojektowany jest z wykorzystaniem szablonów (**template**),
- kontener pozwala na dodawanie i usuwanie obiektów z początku oraz z końca kontenera,
- w określonej instancji kontenera mogą znajdować się obiekty jednego typu,
- kontener nie może mieć z góry określonej maksymalnej liczby zawartych w nim obiektów,
- kontener posiada metodę int size(), która zwraca liczbę elementów w kontenerze
- nie korzystamy z gotowych elementów kontenera z biblioteki STL (takich jak vector, list itp.),
- kontener zawiera przeciążone operatory (dla każdego 1 operator):
    1. (dodanie dwóch kontenerów),
    2. == (porównanie dwóch kontenerów),
    3. = (przypisanie kontenera),
    4. indeksowy (zwraca obiekt znajdujący się pod indeksem i),
    5. << (wypisywania do strumienia informacji o obiektach w kontenerze).
- kontener zawiera metody (dla każdego 1 metoda)

a) void swap(int 1, int m); - funkcja zamieniającą miejscami dwa obiekty
b) int indexOf(const T &t); - funkcja podaje indeks obiektu t w kontenerze
c) bool contains(T &t); - sprawdza, czy obiekt t jest w kontenerze
d) void moveToFront(int numer); - przenosi i-ty obiekt na początek kontenera
e) void moveToEnd(int numer); - przenosi i-ty obiekt na koniec kontenera
f) void removeElement(int index); - usuwa i-ty obiekt z kontenera

Zadanie	**Punkty**
Dziedziczenie	2
Wirtualność	1
Deklaratory zakresu	1
Lista/Kontener/Template	2
Dodawanie/usuwanie/size	2
Metoda/operator	2
Razem	10

8-10			Operator	Metoda
1.	Dąbrowska	Agnieszka	1	f
2.	Golovatyuk	Jana	2	e
3.	Grundland	Paweł	3	d
4.	Kazimierczak	Agata	4	b
5.	Mróz	Jan	5	c
6.	Reda	Paweł	1	a
7.	Tuzinek	Przemysław	2	a
8.	Wojtkiewicz	Szymon	3	e
9.	Wynimko	Jędrzej	4	d
10.	Żurawski	Bartosz	5	f

16-18				
1.	Lorenz	Katarzyna	1	e
2.	Zaczek	Krzysztof	2	f
3.	Radzimirski	Maciej	3	b
4.	Jagielski	Maciej	4	a
5.	Matyjanka	Anna	5	a
6.	Makowski	Jakub	1	d
7.	Wójcicka	Julia	2	c
8.	Jurczuk	Natalia	3	e
9.	Apolinarski	Krzysztof	4	c
10.	Michałowski	Jakub	5	b