Napisać program który będzie symulował działanie segregatora. Segregator przechowuje różne dokumenty uporządkowane wg. zadanych kategorii. Powinien mieć możliwość dodania dokumentu do kategorii a także usunięcia go po zadanym id. Ma też możliwość pokazania wszystkich dokumentów z danej kategorii. Przez dokument rozumiemy jakiś tekst. Kategoria to pojedynczy identyfikator typu string (nie zwracamy uwagi na wielkość liter, można przyjąć że wszystkie kategorie są DUŻYMI LITERAMI). Segregator powinien być zaimplementowany jako klasa posiadająca wygodny interfejs użytkownika. Zadbać o hermetyzację. Następnie napisać klasę pozwalającą budować segregator na bazie wejścia oraz klasę wyświetlająca segregator na wyjście.

Komendy:

+ <kategoria> <tekst> - dodaje tekst w zadanej kategorii, zwraca id tekstu

- <id\_tekstu> - usuwa tekst o podanym id

# <kategoria> - wyświetla teksty i ich id-ki w zadanej kategorii

# - wyświetla wszystkie teksty wraz z ich idkami i podaną kategorią (kategorie w kolejności alfabetycznej)

\*Dodać możliwość edycji tekstu:

@<id\_tekstu> <nowy\_tekst>

\*\*Dodać możliwość zapisu segregatora do pliku xml wraz z możliwością odczytu

> <nazwa\_pliku> - zapis

< <nazwa\_pliku> - odczyt

Zadanie domowe

Segregator 2.0

Przerobić program \*\* tak, żeby tekst nie był już tylko tekstem ale mógł być także plikiem (2 typy danych). Plik różni się od tekstu tym, że nie trzymamy jego treści, ale ścieżkę dostępu. Należy zadbać, aby interfejs użytkownika a) pozwalał podać ścieżkę do pliku, b) wypisywał na ekranie zawartość pliku.

Symulacja

Napisać program symulujący (za pomocą klasy) działanie dowolnego przedmiotu. Np. szafka. Szafkę możemy otworzyć lub zamknąć. Składa się z półek na których znajdują się różne obiekty. Obiekty możemy wyjąć lub schować. Po otwarciu szafki możemy określić ile przedmiotów się w niej znajduje (ale w zamkniętej już nie). Możemy szafkę posprzątać (należy wtedy wyjąć z niej na chwilę wszystkie przedmioty).

Każda operacja powinna mieć logiczny sens. Np. nie możemy wyjąć przedmiotu z zamkniętej szafki. Następnie napisać program symulujący, wykonujący różne działania (albo interfejs użytkownika za pomocą którego będziemy mogli te operacje wykonywać)

Powyższy przykład jest tylko koncepcją, należy pomyśleć nad własnym obiektem. Przykłady: stół, lodówka, suszarka, rower, kubek.