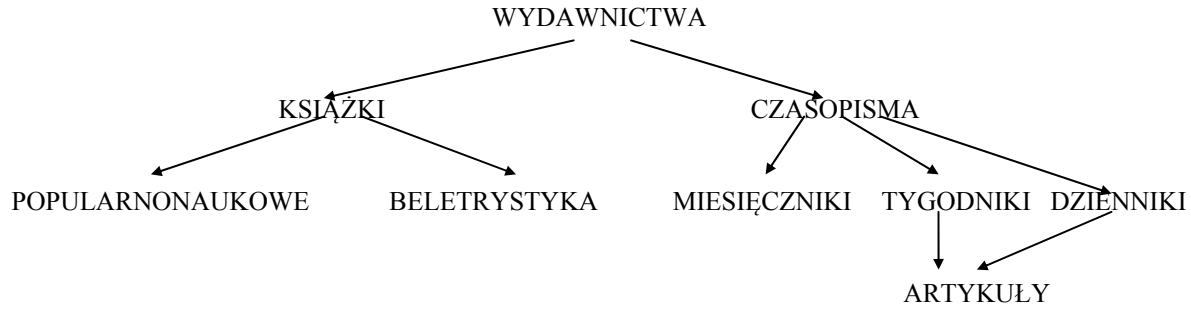
3



Schemat przedstawia hierarchię klas.

Każdy węzeł reprezentuje jedną klasę.

Klasy położone niżej dziedziczą z klas położonych wyżej zgodnie z siecią połączeń

Nazwy klas można modyfikować

Należy:

zdefiniować poszczególne składowe każdej z klas (pola i metody) – nazwy i liczność pół i metod są zależne od osoby realizującej projekt;

określić sposób dziedziczenia klas, czyli zaimplementować przedstawioną hierarchię.

Jako liść rozumiana jest klasa, która nie ma potomka.

Program powinien umożliwiać następujące operacje na zbiorze obiektów:

dodawanie obiektu (tylko do liści);

usuwanie obiektu (tylko z liści);

modyfikacje obiektu;

zapis zbioru do pliku;

odczyt zbioru z pliku;

przeglądanie podzbioru obiektów zgodnie z przedstawioną na rysunku hierarchią - z dowolnego węzła drzewa, wyświetlane zostaną tylko obiekty należące do liści, które z danego węzła dziedziczą, lub w przypadku liścia – jedynie obiekty do niego należące

Każda klasa powinna posiadać co najmniej 1 pole prywatne i 1 pole protected

Poruszanie się w strukturze z linii komend:

1) W systemie znajdują się obiekty.

2) Tylko liść zawiera w sobie listę obiektów. Długość listy jest nieograniczona

3) Struktura - operacje:

- CD [nazwa węzła(klasy)]- zmiana węzła w strukturze

5) Obiekty - operacje:

- MO [obiekt]- tworzy obiekt podany jako parametr dla bieżącego liścia– należy podać parametry obiektu
- DO [obiekt]- usuwa obiekt podany jako parametr dla bieżącego liścia
- MDO [obiekt] – modyfikacja obiektu podanego jako parametr dla bieżącego liścia

6) Polecenie DIR - wyświetla informacje o obiektach widocznych z danego poziomu - domyślnie tylko informacje o nazwach obiektów (wyświetla listę wszystkich obiektów należących do liści, które dziedziczą z danej klasy
Polecenie SHOW [obiekt] – wyświetla szczegółowe informacje o obiekcie
SAVE – zapis zbioru do pliku
READ – odczyt zbioru z pliku
7) Polecenie TREE - wyświetla całą strukturę przedstawioną na rysunku np. w formie wcięć
8) Zapis i odczyt informacji na dysk i konsole ma być zrealizowany za pomocą strumieni.
9) EXIT – wyjście z programu

Implementacja projektu powinna zostać poprzedzona zamodelowaniem problemu za pomocą diagramu klas UML;
Diagram klas UML opisujący zamodelowane klasy powinien zostać wysłany na adres prowadzącego do 04.11.2019 do godziny23:59;

Każda klasa powinna być rozdzielona na dwa pliki: *.cpp oraz *.h (lub *.hpp). W pliku *.h powinny znajdować się tylko deklaracje, a w pliku *.cpp tylko implementacje.

W projekcie powinna znaleźć się co najmniej jedna klasa generyczna, jak również powinno zostać wykorzystane przeciążanie operatorów.

Wyjątek stanowić będzie klasa generyczna, która będzie zdefiniowana w jednym pliku.

Menu powinno być zrealizowane w postaci konsolowej (forma okienkowa wg uznania).

Ewentualne modyfikacje wymagań, własne pomysły na rozszerzenie aplikacji należy konsultować z prowadzącym.

Uwaga: Aplikację da się napisać bez tworzenia klasy generycznej czy przeciążania operatorów, niemniej jednak te elementy są wymagane jako pokazujące niektóre cechy programowania obiektowego.