

Należy stworzyć zestaw klas opisujących statki.

1. Zaimplementować klasę `JednostkaPlywajaca`, będącą klasą bazową dla różnych statków wodnych. Klasa `JednostkaPlywajaca` jest klasą **abstrakcyjną** i przechowuje następujące pola:

nazwa - użyć typu **string**, (np. „QueenMarry”)
cena (cena statku w milionach PLN, domyślnie 50),

Pola dostępne są tylko dla klas potomnych.

W części publicznej klasa zawiera deklarację **wirtualnych metod** `Wypisz()`, `Wczytaj(char* nazwaPliku, int linijka)` oraz `Zapisz(char* nazwaPliku)`. Konstruktor/destruktor w zależności od potrzeb. **1 p.**

2. Jako przykład klasy pochodnej zaimplementować klasę `Statek`, będącą klasą potomną klasy `JednostkaPlywajaca`, zawierającą dodatkowe pola danych danego typu statków (liczba pasażerów, masa (w tysiącach ton), wyporność (w tysiącach ton)) oraz odpowiednie metody. Zdefiniować konstruktor domyślny (wszystkie wartości domyślnie ustawione na 0, konstruktor ten powinien również wywoływać konstruktor klasy nadrzędnej), jak również konstruktor inicjalizujący wszystkie pola klasy `JednostkaPlywajaca` (wraz z nazwą i ceną z klasy nadrzędnej) na wartości podane w argumencie.

- Metoda `Wypisz` powinna wypisać na ekran informację typu: "Statek o nazwie QueenMarry, 150 pasazerow, masa 5 tys. ton, wypornosn 1 tys. ton, cena 20 milionow PLN".
- Metoda `Wczytaj(char* nazwaPliku, int linijka)` powinna wczytać z pliku, którego nazwa podana jest jako jej parametr (`nazwaPliku`), z linii o numerze podanym jako drugi parametr (`linijka`), dane statku oraz przypisać je odpowiednim składnikom klasy `StatekMorski`. Przykład pliku: `StatkiPowietrzne.txt`:
QueenMarry 5000 200 50 120
Orzel 400 100 20 50
Blyskawica 200 50 10 20
gdzie kolejno mamy nazwę, liczbę pasażerów, masę, wyporność oraz cenę.
- Metoda `Zapisz` powinna zapisać do pliku, którego nazwa podana jest jako jej parametr, wszystkie parametry statku morskiego **2 p.**

3. Napisać program, testujący działanie klasy:

- Stworzyć obiekt `ship1` klasy `Statek` (używając konstruktora domyślnego). Wypisać (używając metody `Wypisz`).
- Stworzyć wskaźnik (wraz z obiektem) `ship2` na klasę `Statek` (ustawić jakieś rozsądne wartości parametrów). Wypisać.
- Wczytać do obiektu `ship1` dane z drugiej linii z pliku którego nazwa podana jest jako **pierwszy parametr wywołania programu**. Wypisać. Przykładowy plik należy stworzyć samodzielnie.
- Zapisać dane obiektu `ship2` do pliku podanego jako **drugi parametr wywołania programu**. **1 p.**

4. W funkcji głównej należy stworzyć dynamicznie dwuwymiarową tablicę typu `bool` (tablica powinna mieć wymiary podane jako trzeci i czwarty parametr wywołania programu) – będzie to mapa położenia statków, domyślnie każdy element jest „false”. Następnie należy zmienić element `[0,0]` na „true” i wypisać macierz na ekran, np. dla tablicy 2 x 3 powinniśmy uzyskać na ekranie: **1 p.**

```
1 0 0
0 0 0
```

Dodatkowe: Na kolokwium mają sporą szansę pojawić się również **tablice dwuwymiarowe jako składnik klasy** (w celu przeciwiecia można w tym zadaniu dodać dwuwymiarową – 3 x 3 – dynamicznie alokowaną tablicę `tab` wypełnioną domyślnie liczbą 7, do klasy `Statek`) oraz **losowanie** (liczb całkowitych i zmiennoprzecinkowych, by przeciwiezić można w konstruktorze domyślnym losowanie z rozkładu jednorodnego z przedziału `[3,15]`).

5. Należy stworzyć `Makefile`, umożliwiający kompilację napisanego programu, definiując zmienną `CXX` definiującą kompilator (`g++`) oraz `CXXFLAGS` definiującą flagę (`-Wall`).

Cały program powinien być napisany zgodnie z obowiązującą dobrą praktyką programowania: klasy dzielimy na pliki *.cpp i *.h, powinny być otoczone strukturami ifndef, odpowiednie **wcięcia** powinny być zachowane, (itp). Program musi się kompilować bez żadnych błędów i ostrzeżeń (za niekompilujący się program przewidziana jest kara: tylko 40% zdobytych punktów – maksymalnie można zdobyć 8 p).