Programowanie Obiektowe
Projekt Końcowy

mgr inż. Paweł Trajdos

1 Projekt – Wymagania

1.1 Produkt finalny

Gotowy projekt powinien zawierać:

1. kod źródłowy aplikacji,
2. dokumentację.

1.2 Dokumentacja

Na dokumentację składają się:

1. Uproszczony diagram klas UML,
2. komentarze w kodzie aplikacji.

2 Ocena

2.1 Zasady oceniania

1. Projekt musi być oddany w terminie określonym przez prowadzącego. Za spóźnienie naliczane są kary – patrz wprowadzenie.
2. Każdy student przedstawia swoją pracę prowadzącemu indywidualnie.
3. Wymagana jest znajomość struktury programu oraz funkcjonalności aplikacji.
    - Punkty za poszczególne elementy projektu **nie** zostaną przyznane jeżeli student nie zna sposobu ich działania.
    - Punkty za poszczególne elementy projektu **nie** zostaną przyznane jeżeli student nie potrafi, w obecności prowadzącego, dokonać niewielkich modyfikacji wskazanych fragmentów.

2.2 Kryteria

Każdy projekt **musi** zawierać następujące elementy programowania obiektowego (oczywistym jest, że ich realizacja musi być prawidłowa):

- definiowanie klas;
- hermetyzacja;
- przeciążanie operatorów (jeżeli to możliwe w danym języku programowania);
- dziedziczenie;
- polimorfizm;

Program, który nie zawiera tych elementów nie może zostać zaliczony. Ponadto w projekcie oceniane będą:

- Jakość wykonania:
  - Kod nie powinien zawierać błędów uniemożliwiających kompilację (programy niekompilujące się nie będą oceniane).
  - Zaimplementowane rozwiązania nie powinny zawierać błędów takich jak wycieki pamięci, zapis/odczyt poza przydzieloną pamięcią itp.
  - Kod powinien być podzielony na moduły (tj. każda klasa powinna posiadać własny plik nagłówkowy i plik implementacji).
  - Czytelność kodu (komentarze, formatowanie). Dobry przykład jak **nie** należy formatować kodu można znaleźć tutaj [1].
- Sposób doboru technik projektowania obiektowego do realizowanego zadania.
- Pomysłowość zaimplementowanego rozwiązania.
- Terminowość oddawania zadań:

3 Projekt – Przykładowe tematy

Poniżej przedstawiam przykładowe tematy projektów. Zachęcam jednak do zaproponowania własnych tematów.
Temat 11.0: Ekwipunek
Opracować uproszczony ekwipunek postaci z gry RPG. Założenia:

- Ekwipunek ma ustaloną pojemność.
- Ekwipunek powinien zostać podzielony na „plecak” i „skrzynię”, pomiędzy którymi powinna istnieć możliwość przenoszenia przedmiotów.
- Powinno istnieć kilka klas przedmiotów.

- Przedmioty powinny być opisane atrybutami takimi jak cena, ciężar, stopień zużycia, itp.
- Należy zaimplementować mechanizm zapisujący/odczytujący stan ekwipunku do/z pliku.
- Zaimplementować prosty mechanizm wizualizacji przedmiotów w ekwipunku.

Temat 11.1: Zarządzanie Oddziałami
Zaimplementować zarządzanie oddziałami dla gry typu Strategia turowa(np. Seria HoMM [2], Seria Disciples [3]). Założenia:

- Gracz może zarządzać kilkoma oddziałami.
- Oddziały mogą składać się z wielu jednostek różnych typów.
- Jednostki powinny być opisane współczynnikami takimi jak punkty życia, czy zadawane obrażenia.
- Należy uwzględnić możliwość zdobywania doświadczenia i awansu jednostek.
- Jednostki można wymieniać pomiędzy oddziałami.
- Jednostki można nabywać lub sprzedawać.
- Zaimplementować możliwość rozegrania pojedynku między dwoma oddziałami.
- Zaimplementować mechanizmy wczytywania/zapisywania stanu oddziałów z/do pliku.

Temat 11.2: Uproszczona sieć Petriego
Zaimplementować uproszczony symulator sieci Petriego [4](Implementacja grafów [5]). Założenia:

- Sieć Petriego należy zaimplementować jako dwudzielny, ważony, skierowany graf.
- Oprogramować wczytywanie/zapis grafu z/do pliku.
- Oprogramować metodę wizualizacji grafu.
- Użytkownik powinien mieć możliwość wskazania, które przejście chce odpalić. Jeśli odpalenie przejścia jest możliwe należy zmienić stan sieci.

Temat 11.3: Problem komiwojażera
Zaimplementować zachłanny algorytm rozwiązujący problem komiwojażera(Implementacja grafów [5]). Założenia:

- Schemat połączeń między miastami zaimplementować jako graf nieskierowany.
- Oprogramować wczytywanie/zapis grafu z/do pliku.
- Oprogramować metodę wizualizacji grafu.
- Rozpoczynając od wskazanego miasta znaleźć drogę dla komiwojażera.

Temat 11.4: Wyszukiwanie najkrótszej ścieżki
Dla grafu skierowanego zaimplementuj algorytm odnajdywania najkrótszej ścieżki między danymi wierzchołkami(Szczegóły [5]). Założenia:

- Oprogramować wczytywanie/zapis grafu z/do pliku.
- Oprogramować metodę wizualizacji grafu.
- Dla zadanych wierzchołków wyznaczyć najkrótszą ścieżkę, łączącą te wierzchołki.

Temat 11.5: Obiektowa baza danych
Oprogramować prostą bazę danych (np. biblioteka) przechowującą obiekty rożnych typów. Założenia:

- Obiekty powinny być opisane zbiorem atrybutów i operacji, które można na nich wykonać.
- Zaimplementować mechanizm dodawania/usuwania obiektów w bazie.
- Zaimplementować mechanizm wyszukiwania obiektów w bazie.
- Zaimplementować mechanizm zapisu/odczytu bazy danych do/z pliku.

Temat 11.6: Terminal lotniczy
Oprogramować symulator terminala lotniczego. Założenia:

- Terminal powinien obsługiwać różne typy samolotów (co najmniej 3).
- Należy przechowywać dane o samolotach(stan, położenie, itp.).
- Terminal powinien obsługiwać zarządzanie pasami startowymi.
- Terminal powinien obsługiwać korytarze powietrzne.
- Dodać możliwość zapisu i odczytu danych do/z pliku.

Temat 11.7: Algorytm karuzelowy
Zaprojektować system zarządzania zadaniami oparty na algorytmie karuzelowym. Założenia:

- System powinien zarządzać kilkoma (co najmniej trzema) różnymi typami zadań.
- System powinien przechowywać informacje o zadaniach np postęp, priorytet.
- System powinien mieć możliwość zapisu i odczytu stanu przetwarzania.

Temat 11.8: Gra paragrafowa
Oprogramować system dla gry paragrafowej. Założenia:

- Paragrafy tworzą graf po którym porusza się gracz.
- Oprogramować przynajmniej 3 rodzaje paragrafów.
- System powinien umożliwiać zapisanie i wczytanie stanu gry do/z pliku.

Temat 11.9: Pojazdy Pancerne
System nadzorujący montaż pojazdów pancernych. Założenia:

- Na linii montażowej składane są co najmniej trzy różne rodzaje pojazdów (np. czołg, niszczyciel czołgów, działo samobieżne);
- Każdy pojazd składa się z różnych modułów (silni, działo, wieża);
- Moduły opisane są takimi współczynnikami jak: waga, cena, żywotność;
- System powinien umożliwiać 'składanie' maszyn oraz umieszczenie złożonych maszyn w hangarze.

4 Literatura

[1] The international obfuscated c code contest, `http://www.ioccc.org/years.html`.

[2] Ubisoft, Heroes of might and magic, CDROM (-2015).

[3] S. First, Disciples, CDROM (-2015).

[4] Sieci petriego, `http://jedrzej.ulasiewicz.staff.iiar.pwr.wroc.pl/Progr-Wspol-i-Rozprosz/wyklad/Sieci-Petriego13.pdf`.

[5] T. Cormen, Wprowadzenie do algorytmów, Wydawnictwa Naukowo-Techniczne, Warszawa, 2001.