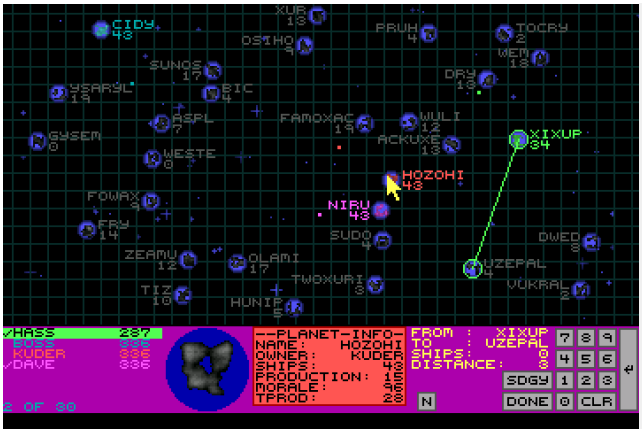
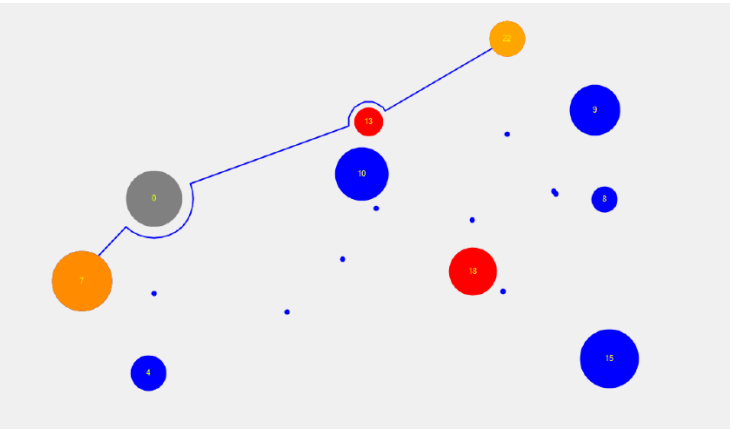
Semestrální práce ICSHP – 2018/2019

Vaším úkolem je realizovat základ jednoduché hry – RTS dobývání vesmíru ve vzoru Galactic Conquest (1987), Galcon, Galcon Fusion, …





Základní požadavky na obsah hry:

- GUI (Windows Forms, WPF, DirectX, OpenGL) se zobrazením mapy, možností myší vybírat planety, určit poměr odesílaných jednotek a výběr cíle
- hráč, neutrální planety, CPU nepřítel
- update smyčka, která každých 20 ms aktualizuje stav hry a způsobí překreslení mapy;
- splnění minimálních požadovaných vlastností hry (viz dále).
- **Semestrální práce musí být vypracována samostatně, není přípustná žádná shoda s ostatními pracemi. Minimálně 1x v průběhu semestru je nutné vývoj SP konzultovat s vyučujícím. SP musí být odevzdána nejpozději v zápočtovém týdnu. SP musí být realizována v programovacím jazyku C#. Použití externích knihoven je možné pouze po předchozí konzultaci.**

Základní principy

- **Svět**
  - hra se odehrává ve 2D prostoru (kartézský souřadný systém)
  - v prostoru se nacházejí planety a létající vesmírné lodě
  - svět nesmí obsahovat konflikty – planety v těsném kontaktu, apod.
- **Planeta**
  - Definována pozicí, velikostí, majitelem (barvou), počtem přítomných jednotek
  - Neutrální planeta nedělá nic

  - Obsazená planeta generuje nové jednotky (předepsanou rychlostí dle velikosti planety)
  - Obsazená planeta může vytvářet jednotky až do dosažení maximálního množství
  - Pokud je počet jednotek na planetě více než přípustné maximum, jednotky postupně umírají

  - Z planety je možné vyslat vesmírnou loď na cílovou planetu
  - Na planetu může přiletět vesmírná loď
  - Každá planeta obsahuje předepsaný počet kontaktních bodů ze/do kterých mohou startovat/přistávat vesmírné lodě
  - Každá planeta obsahuje předepsaný počet obletových bodů, po kterých se pohybují vesmírné lodě, které míří na planetu, která není dostupná po přímé trase

- **Vesmírná loď**
    - Definována pozicí, majitelem (barvou), počtem jednotek na lodi
    - Letí prostorem předepsanou rychlostí
    - Může letět z libovolné na libovolnou planetu

    - Loď preferuje nejkratší přímou cestu
    - Při průletu prostorem musí loď oblétat planety v bezpečné vzdálenosti (definované obletové body planety) kvůli vlivu gravitace.

    - Pokud přiletí na planetu, která má stejného majitele – zvýší počet jednotek na planetě o počet jednotek v lodi
    - Pokud přiletí na neutrální nebo nepřátelskou planetu – sníží počet jednotek na planetě. Pokud by bylo dosaženo záporu (tj. na lodi je více jednotek než na planetě – a zbyde alespoň jedna živá jednotka) – obsadí planetu. Pokud počet jednotek na lodi je stejný jako na planetě, planeta nemění majitele.
- **Hráč**
    - Může kdykoliv vyslat z libovolné (své) planety určité procento jednotek na libovolnou cílovou planetu
- **CPU nepřítel**
    - V náhodných intervalech vybere náhodnou svou planetu a vyšle 50 % jednotek na libovolnou cílovou planetu.

Technické požadavky na realizaci

Realizované typy a jejich složky:

- Herní objekt

- **Abstraktní** předek pro planety a vesmírné lodě
- Obsahuje **virtuální vlastnosti**:
    - Pozice objektu (celočíselné souřadnice x, y)
    - Velikost objektu (celé číslo)
    - Majitel objektu (barva)
    - Příznak dokončení (pokud je platný – objekt bude ze světa odebrán)
- **Abstraktní metoda „aktualizace“**, jako parametr přejímá počet uběhnutých milisekund od poslední aktualizace

- Planeta

- **Potomek herního objektu**
- Obsahuje **vlastnosti**:
    - Počet jednotek (celé číslo)
- Obsahuje **konstanty**:
    - Počet hraničních bodů (32)
    - Násobek poloměru pro určení polohy kontaktních bodů (1.0)
    - Násobek poloměru pro určení polohy obletových bodů (1.4)
- **Metoda aktualizace**:
    - Pokud je planeta neutrální – nedělej nic
    - Pokud je počet jednotek roven maximu (velikost planety * 3)
    - Pokud uplynul stanovený čas – zvyš počet jednotek o jedna
        - ts = velikost <= 32 ? velikost – 8 : velikost
        - aktualizační čas v ms = (Log10(50 – ts) * 10) / 20

- **Metoda „při příletu vesmírné lodě“** s parametrem „vesmírná loď“
  - Pokud majitel vesmírné lodě je stejný jako majitel planety – zvyšte počet jednotek na planetě o počet jednotek na lodi
  - Jinak snižte počet jednotek o počet jednotek na lodi
    - Pokud je počet záporný – planeta mění majitele na majitele lodi
- Veřejná **vlastnost** (pouze pro čtení) **„kontaktní body planety“** (datový typ – kolekce Point)
  - Vrátí kontaktní body planety – určený počet bodů v určeném poloměru (vizte konstanty)
- Veřejná **vlastnost** (pouze pro čtení) **„obletové body planety“** (datový typ – kolekce Point)
  - Vrátí obletové body planety – určený počet bodů v určeném poloměru (vizte konstanty)

- Vesmírná loď

- **Potomek herního objektu**
- Obsahuje **vlastnosti**:
  - Výchozí planeta (planeta)
  - Cílová planeta (planeta)
  - Počet jednotek na lodi (celé číslo)
  - Vypočítaná cesta lodi v prostoru (kolekce Point)
  - Celková již odlétnutá vzdálenost (celé číslo)
- **Metoda aktualizace**:
  - Loď se posune o uplynulý čas [ms] / 8
  - Pokud dosáhne cíle, vyvolá metodu „při příletu na planetu“ nad cílovou planetou a označí příznak „dokončení“ v aktuálním objektu
- **Přetížená vlastnost pozice objektu**:
  - Vypočítá souřadnice lodi v prostoru na základě odlétnuté vzdálenosti a vypočítané cesty lodi

- Hlavní třída (GUI)

- Obsahuje **seznam herních objektů** (planety a vesmírné lodě)
- Obsahuje **časovač**, který co 20 ms provádí aktualizaci všech herních objektů a odebírá dokončené objekty ze seznamu

- Při pohybu kurzoru myši zvýrazněte planetu, která se nachází pod kurzorem
- Umožňuje pomocí levého tlačítka myši vybrat planetu hráče
- Umožňuje pomocí pravého tlačítka myši odeslat stanovené množství jednotek na cílovou planetu
- Umožňuje pomocí kolečka myši volit poměr odesílaných jednotek

- Obsahuje jednoduchého **CPU nepřítele**, který periodicky v náhodných intervalech (300-700 ms) odesílá 50 % jednotek z náhodné planety na náhodnou planetu

- Obsahuje algoritmus pro **vygenerování nového světa**, planety nesmí být umístěny na hraně (nebo za hranou) mapy a nesmí se dotýkat (radius obletové vzdálenosti jednotlivých planet); nageneruje planety různé velikosti (16-48), některé hráče, některé neutrální, některé CPU

- Obsahuje **algoritmus pro výpočet cesty mezi dvěma planetami**
  - 1. otestuje, jestli existuje přímá cesta
    - Spojí každý kontaktní bod 1. planety s každým kontaktním bodem 2. planety
    - Ověřuje konflikt testovaných cest s jinými planetami
    - Pokud existuje alespoň jedna nekonfliktní cesta, vybere nejkratší možnou
  - 2. vytvoří obletovou cestu
    - 1) identifikujte planety v přímé cestě a seřaďte je podle polohy kontaktu
    - 2) postupně vybudujte trasu
      - Od výchozí planety (kontaktní bod) k nejbližšímu obletovému bodu první konfliktní planety
      - Kratší variantu (po / proti směru hodinových ručiček) obletu konfliktní planety
      - Od posledního bodu (obletový konfliktní p./kontaktní bod výchozí p.) pokračujte k dalšímu obletovému bodu nebo rovnou k cílovému kontaktnímu bodu

Struktura zdrojového kódu

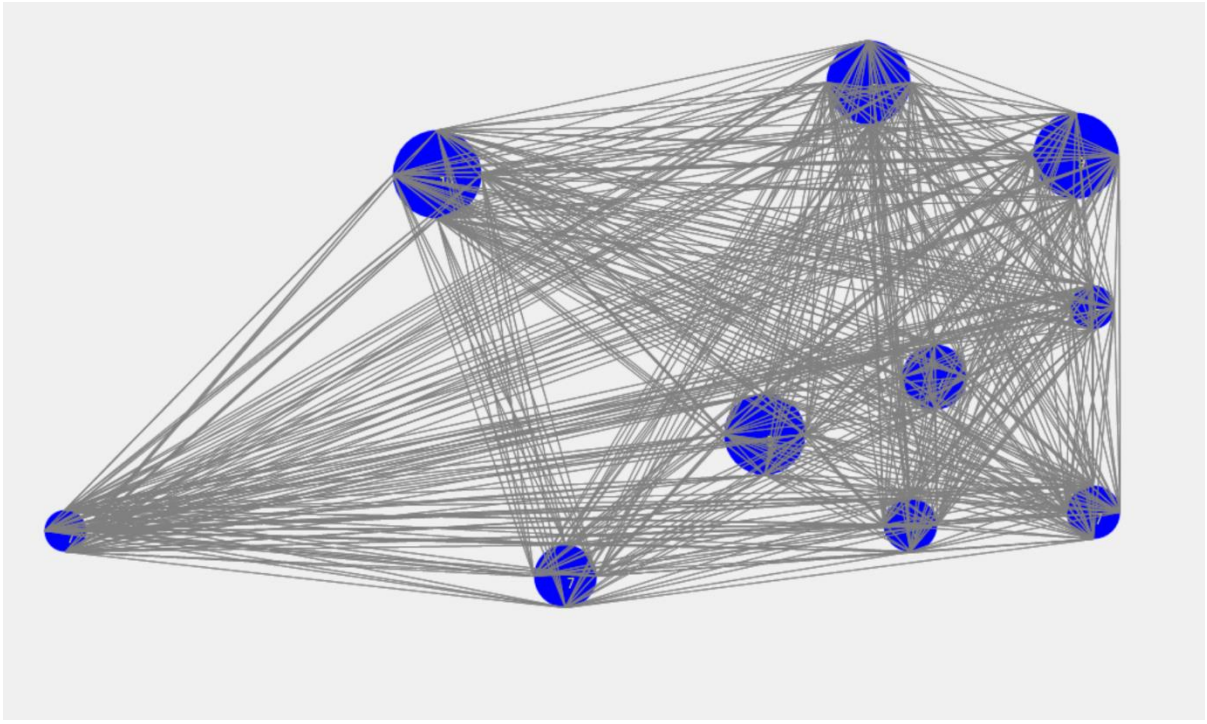
- Kód by měl být srozumitelně logicky strukturován, tj. do oddělených souborů. Není přípustné odevzdat vše v jediném souboru.
- Kód musí splňovat standardní náležitosti a zvyklosti jazyka C# ([1], [2])
- Kód musí dodržovat pravidla a zásady objektově orientovaného programování

Volitelné modifikace

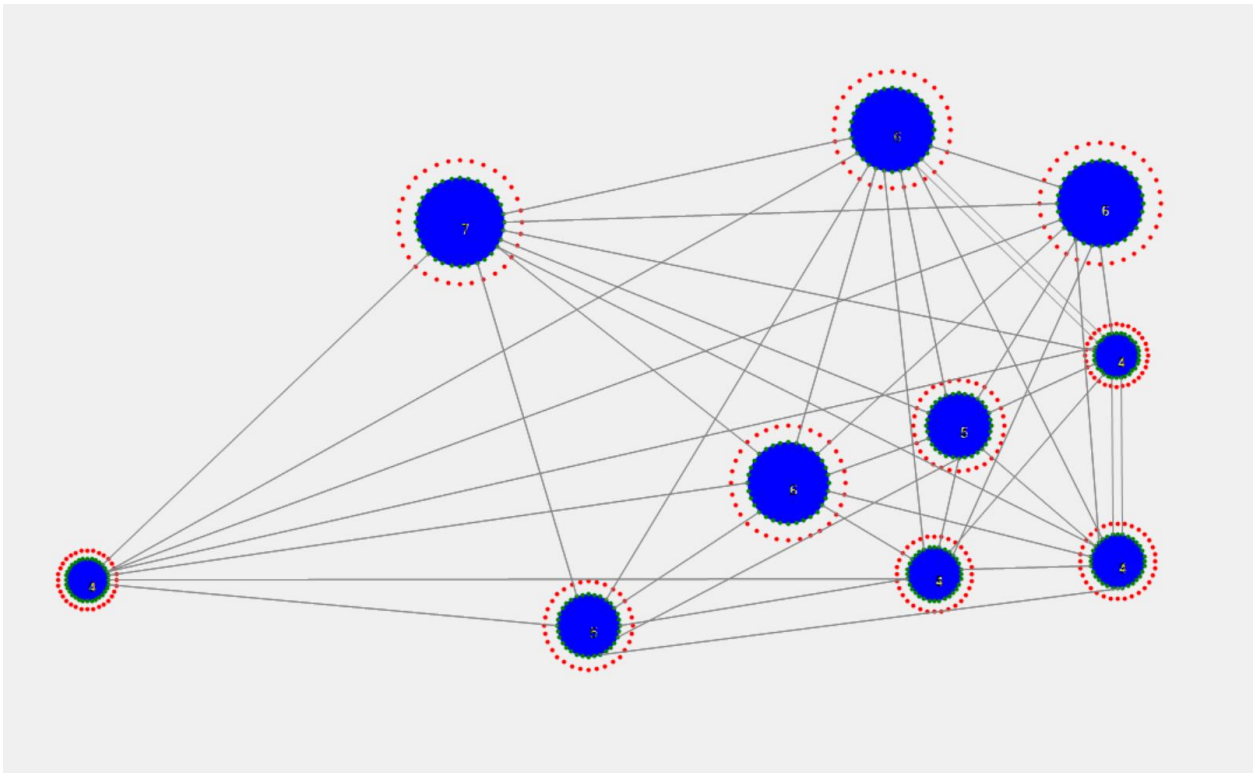
- **realizace modifikací bude odměněna plus body do zápočtu a zkoušky, příklady možných modifikací:**
- hráč může vybrat více planet (tažením myši / shiftem / controlem) a ze všech odeslat dané množství jednotek
- lepší vizualizace – lepší znázornění směru letu lodí, znázornění shluku lodí, znázornění explozí …
- optimalizace letových tras – odeberte nadbytečné body z letové trasy a najděte kratší přípustnou trasu
- inteligentní CPU – CPU chytře vybírá planety, ze kterých bude vysílat jednotky a snaží se o agresivní/defensivní strategii
- možnost střetu létajících lodí ve vesmírném prostoru
- uložení a načtení (save, load) stavu rozehrané hry
- …
- **příklady modifikací vedoucích k <u>velmi výrazným</u> plus bodům:**
- umožnění síťové hry s jiným hráčem
- optimalizace letových tras a jejich výpočet pomocí teorie grafů a algoritmů Floyd/Dijkstra/A*
- záznam průběhu hry (replay) a jeho zpětné přehrání (volitelně pauza, změna rychlosti přehrávání, …)
- opengl / directx vizualizace
- …

Odevzdání projektu

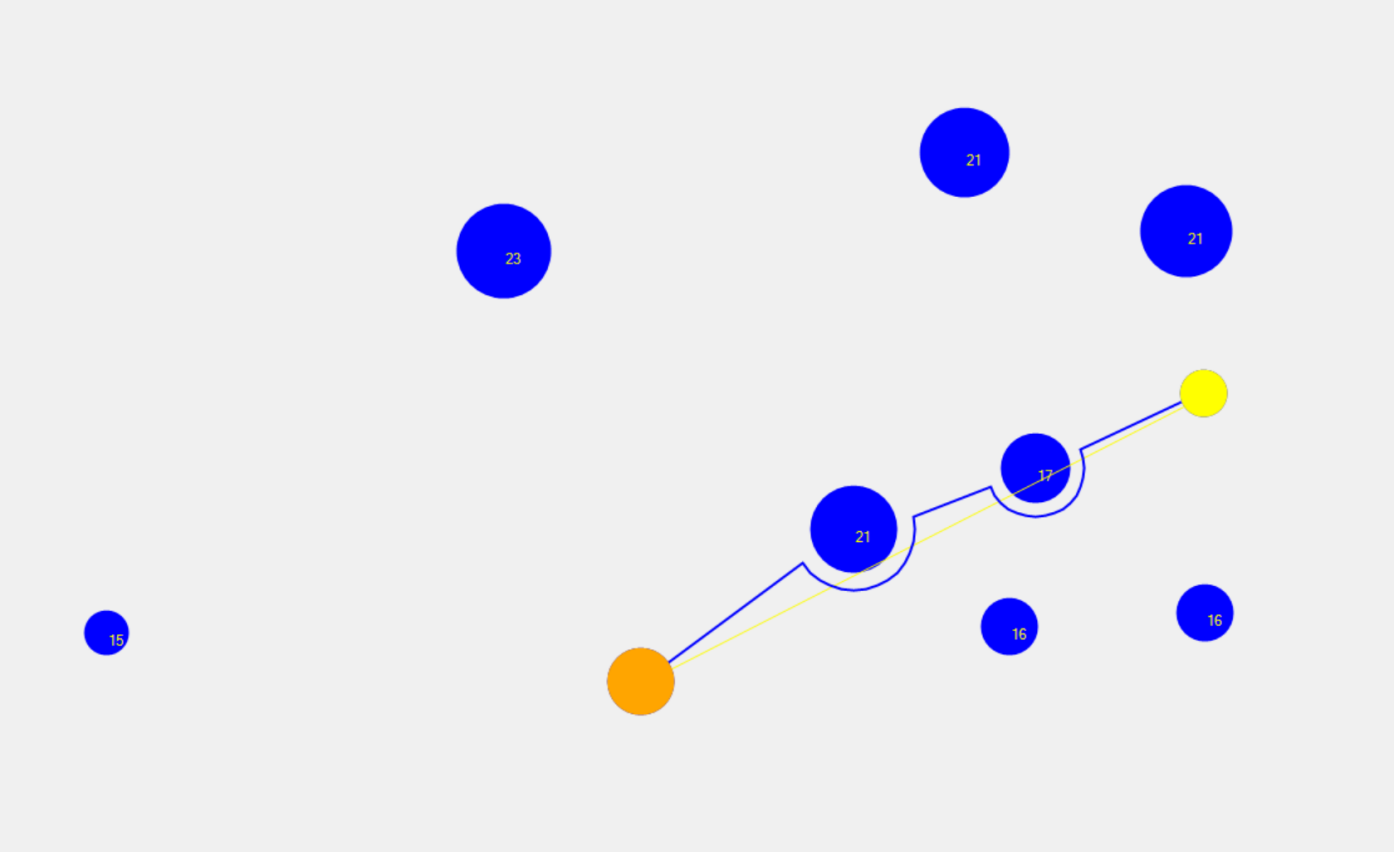
- celý projekt odevzdejte do STAGu ve formátu ZIP
- deadline – konec zápočtového týdne – 3. 5. 2019 23:59
- před odevzdáním projektu odstraňte veškeré binární (spustitelné) soubory
- projekt, který nesplňuje uvedené zadání (tj. minimální požadavky) nebude akceptován



Obrázek 1 - Ukázka rozmístění planet a kontaktních bodů (bez řešení kolizí, pro názornost snížen počet kontaktních bodů na 4)



Obrázek 2 - Ukázka kontaktních a obletových bodů, na obrázku jsou dále vykresleny všechny cesty použitelné pro přímé lety (bez obletů jiných planet)



Obrázek 3 - Ukázka rozdílu mezi přímou a výslednou obletovou trasou