WWW9 - Projekt zespołowy
Zadania kwalifikacyjne

Jakub Sygnowski

29 maja 2013

Info

Kod do zadań 1 i 2 należy przysłać do repozytorium gita: link

Konkretnie, należy zrobić forka danego repozytorium i zrobić w nim folder o nazwie = imie-nazwisko (pod imie, nazwisko podstawić *swoje* imię i nazwisko), napisać w nim kod do zadania 1 i do zadania 2 przed modyfikacją i zrobić pull request albo samemu zrobić merge. Można (i warto) korzystać z gita jako systemu kontroli wersji trzymając tam częściowe wyniki swojej pracy. Następnie, należy napisać modyfikację do zadania 2 i ją wysłać (git push) do repozytorium, rozwiązując konflikt i ponownie zrobić merge'a do mastera.

(Powyższa instrukcja może wydawać się niezrozumiała, dopóki nie pozna się odrobinę gita)

Po wykonaniu powyższych kroków należy wysłać do mnie maila (na `mailto:sygnowski@gmail.com`) z numerami (dwóch) commitów, w których znajduje się - w pierwszym piewrsza wersja zadania 2 (i być może pierwsze zadanie), a w drugim pierwsze zadanie i drugie po modyfikacji oraz rozwiązanie zadania 3. (Tzn. mail zawiera trzy rzeczy).

Rozwiązania można przysyłać do 7 lipca (godzina 23:59:55) 2013. Zachęcam jednak do wysłania ich wcześniej - będę starał się oceniać je na bieżąco i pozwalał poprawiać rozwiązania. Czas mojej odpowiedzi można oszacować z góry przez tydzień, ale myślę, że średnio to będzie ok. 2 dni.

Ocenianie

Za każde z zadań można uzyskać 0, 1 lub 2 punkty (czyli łącznie od 0 do 6 punktów). Aby zakwalifikować się na warsztaty należy zrobić zadania 1 i 2 (z ewentualnie drobnymi błędami), zadanie 3 jest niejako 'nieobowiązkowe', ale zachęcam do zrobienia go, żebyśmy mieli co potem pisać na WWW :P

Jako że kod wszystkich uczestników będzie dostępny publicznie, proszę o nie zaglądanie do kodów kolegów (a tymbardziej o niekopiowanie kodów kolegów :P)

Zadanie 1 - na poprawność[1]

Zaimplementuj klasę Deque przechowującą liczby całkowite (int).

[1] Autorem zadania jest mój prowadzący PO - Maciej Weksej

Klasa ma udostępniać następujące (publiczne) metody:

- `void pushLeft(int el)`
  wstawienie elementu z lewej strony
- `void pushRight(int el)`
  wstawienie elementu z prawej strony
- `int popLeft()`
  usunięcie i zwrócenie elementu z lewej strony
- `int popRight()`
  usunięcie i zwrócenie elementu z prawej strony
- `int at(int pos)`
  zwrócenie elementu na pozycji pos
- `int size()`
  zwrócenie aktualnej liczby elementów w kolejce
- `boolean isEmpty()`
  informacja, czy kontener jest pusty
- `int hasCycle()`
  opis poniżej

Dane mają być przechowywane w zwykłej tablicy. Gdy miejsce w tablicy się kończy, przydzielamy tablicę dwa razy większą większą i przepisujemy zawartość starej tablicy do nowej. Gdy zajętość tablicy spada do $1/4$, przydzielamy tablicę o rozmiarze o połowę mniejszym i przepisujemy zawartość starej tablicy do nowej. Przepisywanie całej tablicy może nastąpić tylko i wyłącznie w tych dwóch sytuacjach.

W sytuacji nieprawidłowego wywołania (np. popLeft() na pustej strukturze czy at(-1)) metoda ma zwracać wartość -1.

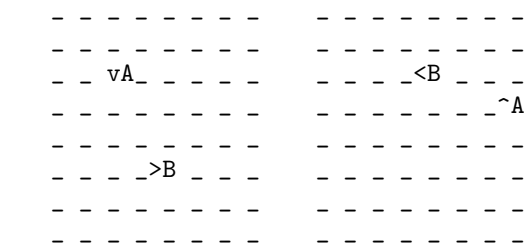
Metoda int hasCycle() ma sprawdzić, czy ciąg $t[0] \to t[t[0]] \to t[t[t[0]]] \to \ldots$ w pewnym momencie się zapętla. Jeśli tak, metoda ma zwrócić długość cyklu (po ewentualnym pominięciu prefiksu), wpp. (np. kiedy $t[0] >= .size()$) ma zwrócić -1. Metoda hasCycle ma nie niszczyć danych zawartych w tablicy.

Zadanie 2 - na projekt[2]

Wersja 1

Na **polu bitwy** odbywa się bitwa robotów. Pole bitwy jest cykliczną (tzn. w kształcie torusa - każde pole ma 4 sąsiadów) planszą o wielkości 8 na 8. Uczestniczą w niej dwa roboty, których celem jest zniszczenie tego drugiego. Przy rozpoczynaniu gry roboty są ustawiane w losowym miejscu planszy, z losowym **obrotem**. Roboty nazwyają się A i B i mają po 5 punktów wytrzymałości.

[2] A to z kolei znalazłem (w trochę innej wersji) na stronie p. Kozubek



Powyższe plansze pokazują cztery możliwe obroty robotów.

Każda tura bitwy składa się z pięciu (gdzie 5 jest ustaloną stałą, ma być możliwość jej łatwej zmiany) akcji wykonywanych przez roboty. Przed turą robot planuje sekwencję 5 akcji, a następnie akcje są na zmianę wykonywane (rozpoczynając od robota A). Rodzaje akcji:

- strzał (S) - robot strzela przed siebie laserem mającym zasięg wzdłuż całej linii (cyklicznie)
- uderzenie(U) - robot wykonuje zamach ramieniem, obejmujący trzy pola przed nim: jedno na wprost i dwa na lewy i prawy ukos
- krok naprzód(+) lub do tyłu(-) - wykonanie go jest możliwe tylko, gdy pole docelowe jest wolne, w przeciwnym wypadku akcja przepada
- obrót - zmiana kierunku zgodnie z ruchem wskazówek zegara (P) lub przeciwnie (L) o 90 stopni

Akcje za wyjątkiem obrotu nie zmieniają kierunku robota. Jeżeli w wyniku S lub U w zasięgu znalazł się przeciwnik, to jego wytrzymałość maleje o 1.

Gra kończy się w momencie, gdy wytrzymałość któregoś robota spadnie do 0.

Roboty mogą planować walkę wg. różnych strategii. Należy zaimplementować jednego robota wykonującego losowy ruch i jednego z własną, sensowną strategią (ale nie należy się za bardzo nad nią przychylać, np. patrzenie, czy można strzelić albo uderzyć innego robota w jednym ruchu i robienie tego albo losowego ruchu jest ok). Oba roboty znają stan całej planszy.

Należy napisać klasę Animacja, która będzie symulować grę i wyświetlać co kilkaset ms (ma to być łatwo zmienialne w kodzie) stan planszy + ilość punktów wytrzymałości robotów (oraz klasy pomocnicze). Animacja ma mieć metodę main.

To jest zadanie na zaprojektowanie swojego kodu - nie musi on działać szybko, być ciekawy pod względem algorytmicznym, ale ma być możliwość łatwego dopisania robota o innej strategii, gry dla więcej niż dwóch robotów itd. Należy użyć dziedziczenia.

Modyfikacja

Po zrobieniu pierwszej części zadania, należy zmienić kod tak, żeby symulował grę, w której każdy robot ma tyle akcji w turze, ile wynosi jego aktualna ilość punktów wytrzymałości. Na przykład, jeśli roboty A i B mają odpowiednio na początku tury 5 i 3 punkty życia, to tura wygląda tak: ABABABAA.

Zadanie 3 - własny pomysł

W tym zadaniu należy przedstawić pomysł na to, co chciałoby się pisać w ramach projektu na zajęciach. Należy przedstawić krótki opis projektu (najlepiej w *.pdf), mile widziany będzie podział na 3-4 moduły/grupy. Szacuję, że w pisaniu projektu weźmie udział ok.10 osób.