LS 2010/11 Pexeso

Domácí úkol I.
V Java EE implementujte pomocí servletů, JSP a Java Beans hru pexeso s volitelným počtem hracích karet (alespoň 4×4, 8×8). Hru hraje vždy jeden hráč proti počítači (algoritmus vizte níže). Obrázky na kartách můžete nahradit čísly. Po dobu sezení se ukládá do žebříčku hráčů počet výher každého hráče a podle počtu výher lze žebříček řadit.

Pravidla hry
Balíček se skládá ze sudého počtu, nejčastěji 64 karet, které tvoří dvojice. Cílem hry je nalézt co nejvíce dvojic. Karty se zamíchají a rozloží lícem dolů tak, aby žádný z hráčů neznal rozložení karet. Hráči postupně otáčí dvojici karet lícem vzhůru, aby je viděli i ostatní hráči. Pokud karty patří k sobě, hráč je odebere a otáčí další dvojici. Pokud karty k sobě nepatří, otočí je zpět lícem dolů a pokračuje další hráč v pořadí. Hraje se tak dlouho, dokud nejsou všechny karty rozebrány. Vítězem se stane hráč s největším počtem nalezených dvojic.

Algoritmus počítačového hráče
Algoritmus je docela naivní, pokud vymyslíte jiný a budete schopni zdůvodnit, že lépe simuluje rozhodování člověka, neváhejte jej implementovat jako druhý - můžete tímto způsobem získat body za excelentní řešení.

Pro simulaci počítačového hráče použijte následující algoritmus. Rozlišujte tři úrovně kvality počítačového hráče rozlišené koeficientem p:

- *low* - p = 0.3
- *medium* - p = 0.6
- *high* - p = 0.9

Předpokládejme, že:

- pozice hracího pole jsou indexovány dvojicí čísel x,y, kde x je pozice na vodorovné ose a y na svislé.
- každá (shodná) dvojice hracích karet má přiřazen unikátní identifikátor.

Počítačový hráč si udržuje datovou strukturu, do které si postupně ukládá na jaké pozici byla nalezena karta s jakým identifikátorem (ukládají se všechny otočené karty - tedy i dalšími hráči!). Např.

Identifikátor	Pozice 1	Pozice 2
1	2,3	
5	3,3	3,2
9	4,3	4,4

Když je hráč na tahu postupuje následovně:
Nejprve vygeneruje náhodné číslo n z intervalu 0,1. Dále postupuje podle porovnání hodnoty n s hodnotou p:

Otočení první karty

1. Podmínka: n <= p
   Hráč z datové struktury vybere takovou kombinaci, která mu zaručuje úspěch v tomto tahu (tj. je známa pozice obou karet). Tyto karty otočí a pokračuje dalším tahem. Algoritmus končí. Pokud takovou kombinaci nezná, *otočí náhodně zvolenou kartu*.
2. Podmínka: n > p
   Hráč *otočí náhodně zvolenou kartu*.

Otočení druhé karty

1. Podmínka: n <= p
   Podle otočené první karty se hráč pokusí nalézt v datové struktuře kombinaci, která by mu zajistila úspěch v tomto kole. Pokud takovou nezná, *otočí náhodně zvolenou kartu*.
2. Podmínka: n > p
   Hráč *otočí náhodně zvolenou kartu*.

Otočení náhodně zvolené karty

1. Podmínka: n <= p
   Pokud je to možné (pokud již nebyly otočeny všechny karty) hráč otočí kartu náhodně, ale vybírá pouze z těch, které ještě nebyly otočeny.
2. Podmínka: n > p
   Karta k otočení je vybrána zcela náhodně.

Hodnocení

Hodnocení domácí úlohy

Část	Body	Poznámka
Třívrstvá architektura	3	servlet(y), JSP, JavaBeans
Funkční session	1	s podporou i bez podpory cookies
Validní (X)HTML, CSS, JavaScript	1	bez chyb a varování
Správná implementace algoritmu počítačového hráče	2	
Obrázkový rub karet s možností nahrání a volby vlastního tématu	1	
Excelentní řešení	2	obrázkové karty s možností nahrání vlastního tématu, další algoritmus počítačového hráče
Celkem	10	