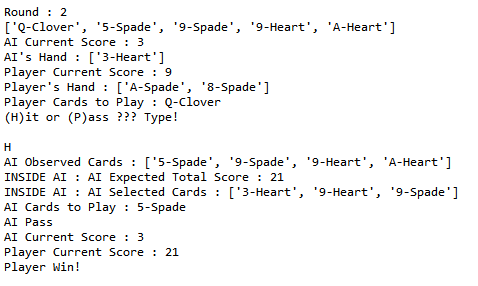
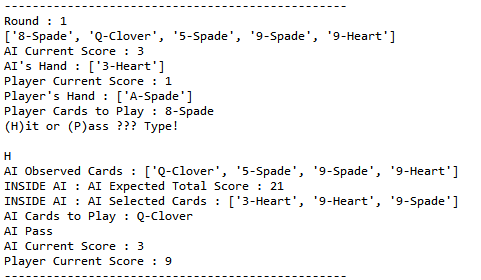
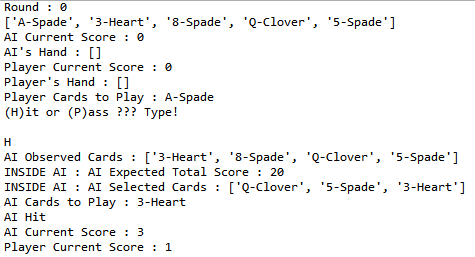
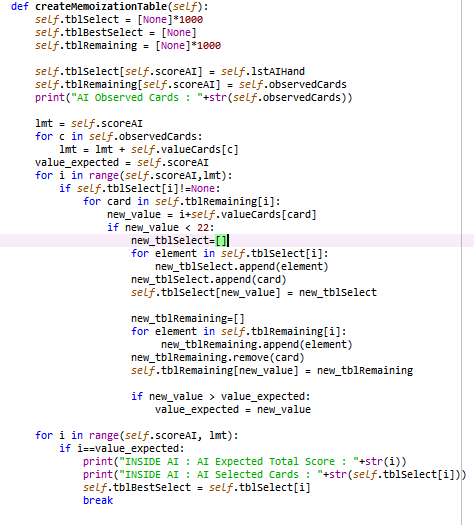
Blackjack with Dynamic Programming

20150669 전예일

Screenshot :



위와 같이 프로그램이 정상적으로 작동하는 것을 확인해 볼 수 있다. AI는 observed card에서 자신이 최고의 점수를(21점을 넘지 않으면서 가장 가까운) 얻을 수 있는 card set을 분석하고 그 set에 적합한 card를 Hit하는 전략을 취하며 게임이 끝날 때까지 진행된다.



위의 screenshot은 memorization 부분의 code이다. 핵심이 되는 내용은 ppt에도 설명되어 있는

self.tblSelect, self.tblBestSelect, self.tblRemaining 이다. Code의 진행은 self.tblSelect는 처음에 None이라는 element로 가득 찬 list에서



라는 문구를 통해 index가 self.scoreAI인 element에 self.lstAIHand를 지정하고,

self.tblRemainning은 마찬가질 처음에 None으로 가득찬 list에서

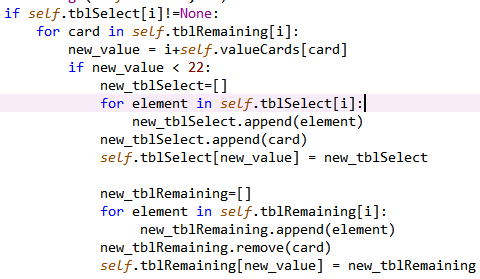


라는 문구를 통해 index가 self.scoreAI 인 element를 self.observedCards를 지정해 준다.

Self.tblSelect는 현재 스코어 i에서 AI의 Hand에 추가되어 있는 card set을 의미하며 self.tblRemaining은 현재 스코어 i에서 AI의 Hand에 추가될 수 있는, observed cards에 남아있는 card들을 의미한다.

그리고 프로그램이 진행되면서 점수에 해당하는 i를 index로 하여 그 지점에서 self.tblRemaining 에 남아있는 card들을 각각 추가하며 그 때, card의 점수만큼 증가한 index에 새로운 self.tblSelect와 self.tblRemaining값이 갱신되게 된다. 즉, 카드 X가 추가되는 상황에서 현재 score에 해당하는 i에 X가 추가된 i+X에 해당하는 index 에서 self.tblSelect에는 card X를 추가한 card set이 갱신되고 self.tblRemaining에는 card X가 remove된 card set이 갱신되게 된다. 다음은 이러한 진행에 대한 code 내용이다.

 에서 index가 최대 진행될 수 있는 limit를 observed card와 현재 AI의 score의 점수의 합으로 정해준다. 그리고 for문을 index i가 현재 AIscore에서 lmt 까지 진행하게 한다.

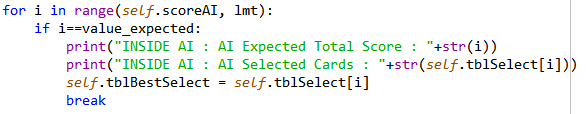


Self.tblSelect[i]가 None인 구간은 현재 card set에서 도달할 일이 없는 구간이기에 진행부분에서 제외하였다. ( if self.tblSelect[i]!=None)

for card in self.tblRemaining[i]: 에서 현재 Remaining에 남아있는 card 각각을 추가하는 모든 경우를 실행하고 new\_value, new\_tblSelect, new\_tblRemaining 이라는 변수를 사용하여 각각의 정보를 갱신한다.

또한 라는 변수를 사용하여 Bestselect를 찾는다. 매번 score에 해당하는 index값을 갱신할 때

 라는 구문을 이용하여 가장 높은 점수 값이 기록되도록 한다. (실행 과정에서 new\_value < 22일 때만 진행하기에 21점을 초과하는 경우는 배제된다.) 실행이 완료된 뒤에는 value\_expected 값에 기록된 self.tblSelect를 self.tblBestSelect에 기록하여 AI의 action을 결정하게 된다.



( BestSelect를 결정하는 code이다.)