과제 1번

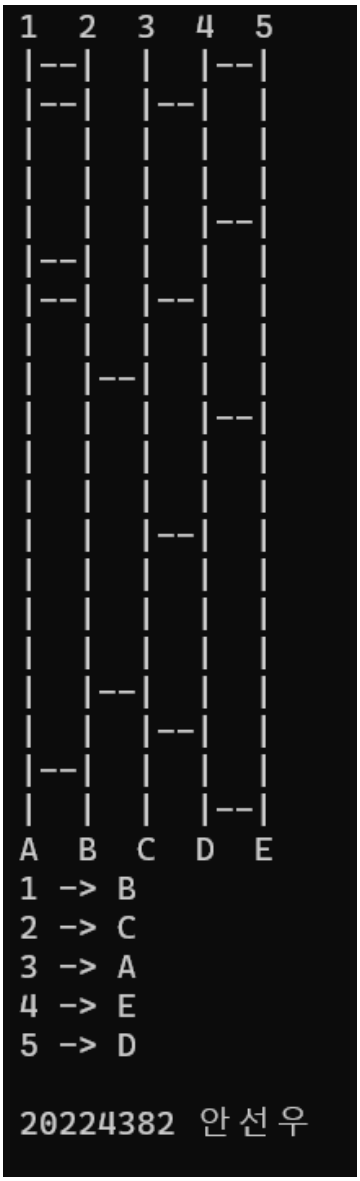
< 사용 변수 >

- char 자료형의 I, j, r 변수는 for 문의 제어변수이며, linecount는 생성된 가로선의 개수를 세는 변수, minline은 최소 가로선의 개수를 의미한다(실습에서는 15개로 지정).
- char 자료형의 ladder[20][4] 배열은 사다리 가로선 저장 배열로, result[5]={65,66,67,68,69} 배열은 사다리 결과 저장 배열로 사용했다.

< 풀이 방법 >

[main 함수]

- while 문을 사용해 linecount가 minline에 도달할 때까지 가로선 생성을 계속하도록 했다. 여기서 rand를 각 배열 요소마다 사용하는 것이 아니라, 배열의 열과 행에 대해서 rand를 한 후, 뽑힌 위치에 가로선이 존재하지 않으며, 양옆에 가로선이 없고 위로 2개의 가로선이 존재하지 않는(몰려서 나타나는 것을 막기 위해) 경우에만 가로선을 확정했다. 이후 ladderresult 함수로 결과를 확정, 이후 사다리를 출력했다. 여기서 삼중 for 문을 사용해 가로선이 두 번 출력되도록 하여 가독성을 높였다. 마지막엔 각 플레이어에 대한 결과를 표시했다.



[ladderresult 함수]

- void 형 함수로, ladder 정보와 result 정보를 받은 후, 밑에서부터 위로 올라가며 플레이어의 결과를 확인한다. 여기서 가로선의 존재에 따라 result 배열의 순서가 바뀌는데, 이를 위해 char 자료형의 saveres 변수를 사용했다.

** 코드 한계 : 가로선은 15개까지만 존재하게 된다. 그 이상 생성하기 위해선 minline의 개수를 수정해야 한다.

< 다른 풀이 방법 >

- rand()를 배열 80칸에 모두 적용할 수도 있다. 그러나 비효율적이다.
- result 배열을 사용하지 않고 줄마다 결과를 측정하는 방법도 있다. for문을 적절하게 사용하면 된다.
- 결과 확인에서도 나는 밑에서 위로 올라가는 방식을 선호하나, 위에서 아래로 내려가는 방식으로도 가능하다. 그러나 이 경우, 결과 마지막에 나오는 양식을 맞추기 위해서는 정렬이 필요하다.

과제 2번

< 사용 변수 >

- short 자료형의 speresult[5][5] 배열은 전체 결과를 저장하는 함수이며, 0으로 초기화했다. char 자료형의 l, j, r 변수는 for 문의 제어변수이며, minline 변수는 사다리가 가져야 하는 최소 가로선의 개수를, short 자료형의 try는 사다리 게임의 시행 횟수를 의미한다. while 문 안에 사용된 char 자료형의 result[5] 배열은 게임의 결과를 잠시 저장해두는 역할이다.
- laddermaker 함수 중 char 자료형의 ladder[20][4] 배열은 사다리 정보를 저장, linecount 변수는 생성된 가로선의 개수를, l, j는 for문의 제어변수이다.
- ladderresult 함수 중 char 자료형의 l, j 변수는 for 문의 제어변수이며, saveres 변수는 result 변수 요소 순서 변경을 위한 중간 저장 역할을 한다.

< 풀이 방법 >

[main 함수]

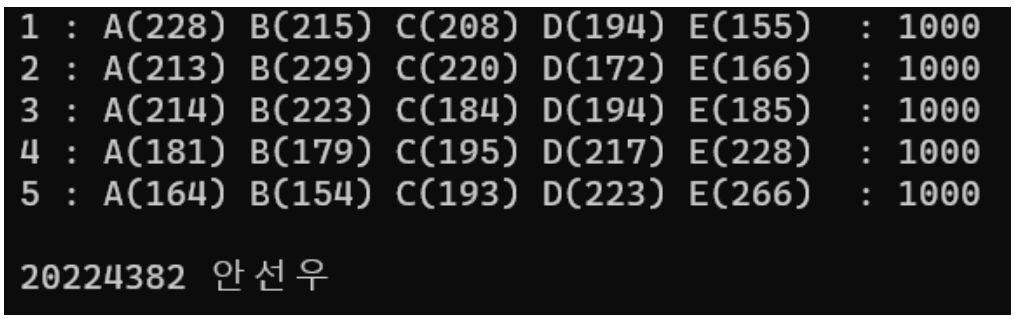
- while 문 안에서 laddermaker 함수 실행 후 speresult 배열에 개별 결과를 저장했다. 매 시행 전 result 함수를 abcde 순서로 수정했다. while 문 종료 후 for 문을 이용하여 전체 결과를 출력했다.

[laddermaker 함수]

- void 형 함수로, while 문을 사용해 linecount가 minline에 도달할 때까지 가로선 생성을 계속하도록 했다. 여기서 rand를 각 배열 요소마다 사용하는 것이 아니라, 배열의 열과 행에 대해서 rand를 한 후, 뽑힌 위치에 가로선이 존재하지 않으며, 양옆에 가로선이 없고 위로 2개의 가로선이 존재하지 않는(몰려서 나타나는 것을 막기 위해) 경우에만 가로선을 확정했다. 이후 ladderresult 함수로 결과를 확정했다.

[ladderresult 함수]

- void 형 함수로, ladder 정보와 result 정보를 받은 후, 밑에서부터 위로 올라가며 플레이어의 결과를 확인한다. 여기서 가로선의 존재에 따라 result 배열의 순서가 바뀌는데, 이를 위해 char 자료형의 saveres 변수를 사용했다.



과제 3번

< 사다리가 공정하지 않다고 생각한 이유 - 가설들 >

- (X) 사다리 열 마다 적어도 2개 이상의 가로선이 존재한다면 결과가 다르게 나타날지 확인해 보았다. 그러나 큰 차이가 없이 계속해서 쏠린 결과가 나왔다. 3개 이상의 가로선이 존재하도록 했을 때도 동일했다.

```
1 : A(248) B(261) C(198) D(154) E(139)  : 1000
2 : A(240) B(221) C(202) D(182) E(155)  : 1000
3 : A(187) B(192) C(203) D(207) E(211)  : 1000
4 : A(153) B(164) C(194) D(233) E(256)  : 1000
5 : A(172) B(162) C(203) D(224) E(239)  : 1000
```

- (O) 사다리 가로선의 개수가 15개 이상이라면 어떤 결과가 나올지 확인해 보았다. minline이 20일 때가 15일 때 보다 비교적 고른 결과가 나옴을 확인할 수 있다, 해당 가설을 더욱 확인해 보기 위해 ladder의 행을 25일 때 minline이 30인 경우를 시행해 보았다. 두 시도 모두 행이 20, minline이 15일 때보단 비교적 공정해졌다. 그럼에도 아직 17%대의 확률이 존재한다는 점에서 한계를 가진다.

```
1 : A(216) B(217) C(213) D(183) E(171)  : 1000
2 : A(223) B(242) C(203) D(181) E(151)  : 1000
3 : A(212) B(208) C(180) D(191) E(209)  : 1000
4 : A(170) B(177) C(202) D(224) E(227)  : 1000
5 : A(179) B(156) C(202) D(221) E(242)  : 1000
```

↑ 행 20, minline 20 / ↓ 행 25, minline 30

```
1 : A(202) B(205) C(209) D(193) E(191)  : 1000
2 : A(203) B(219) C(190) D(197) E(191)  : 1000
3 : A(210) B(210) C(179) D(196) E(205)  : 1000
4 : A(201) B(178) C(227) D(182) E(212)  : 1000
5 : A(184) B(188) C(195) D(232) E(201)  : 1000
```

- (X) 시간을 조작하는 경우도 확인해 보았다. sleep 함수를 사용했다. 처음 조건(행 20, minline 15)을 전제로 하고 Sleep(2)를 한 결과는 이러하다. 크게 차이가 없을 확인할 수 있다. 따라서 sleep 함수를 사용하는 것은 좋은 해결방안이 아니다. (srand 함수가 시간을 시드 값으로 받는다는 점에서 애초에 의미가 없는 가설이다.)

```
1 : A(227) B(241) C(224) D(169) E(139)  : 1000
2 : A(256) B(198) C(207) D(168) E(171)  : 1000
3 : A(208) B(211) C(190) D(209) E(182)  : 1000
4 : A(167) B(173) C(182) D(225) E(253)  : 1000
5 : A(142) B(177) C(197) D(229) E(255)  : 1000
```

< '가로선 개수 늘리기' 가설 채택 이유 및 수정한 코드 결과 >

- 여태까지의 시도 중 가로줄의 개수를 늘리는 경우가 가장 균일한 비율로 결과를 보임을 확인할 수 있었다. 그 이유는 가로선은 ABCDE 결과로 갈 수 있도록 하는 통로의 역할을 하는데, 그 수가 적다면 높은 확률로 일부 결과로만 갈 수 있게 된다. 그렇기에 가로선 개수가 많을수록 다양한 결과로 갈 수 있게 되어 그 확률이 비슷해지는 것이다. 그렇기에 ladder의 배열을 [40][4]로 늘리고, minline을 33으로 설정했다(minline 34 이상부턴 프로그램 진행 시간이 길어져 33으로 설정했다). 그 결과는 다음과 같다.

```
1 : A(199) B(204) C(200) D(190) E(207)  : 1000
2 : A(198) B(214) C(198) D(199) E(191)  : 1000
3 : A(209) B(182) C(200) D(213) E(196)  : 1000
4 : A(207) B(194) C(195) D(186) E(218)  : 1000
5 : A(187) B(206) C(207) D(212) E(188)  : 1000

20224382 안선우
```

- 프로그램 진행 시간제한이 없다는 가정하에 가로선의 개수를 더 늘린다면 더욱 공정한 사다리가 되리라 본다.