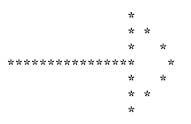
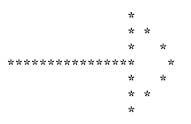
Homework #03은 abstract class ShapeBase를 상속받는 두 클래스 RightArrow와 LeftArrow를 만드는 것을 목표로 한다. RightArrow와 LeftArrow는 drawHere() 메서드를 통해 다음과 같은 그림을 콘솔창에 나타낸다.



두 클래스는 꼬리의 길이를 저장하는 tail과 arrowhead의 길이를 저장하는 width 멤버 변수를 가진다. width는 무조건 홀수여야 하며 두 클래스는 이 유효성을 확인해야만 한다. 이를 검사하기 위해 두 클래스는 checkEven()이라는 메서드를 가진다. checkEven(int number) 메서드는 number가 짝수일 때 true를 리턴하고 홀수일 때 false를 리턴하는 메서드다. checkEven은 두 클래스의 인자가 주어진 생성자가 호출되거나 setOffset() 메서드가 호출될 때 width에 값을 저장하기 전, 유효성을 확인하기 위해 호출되어진다. 만약 width가 짝수라면 “Width must be odd!”를 콘솔창에 출력하고 생성자와 setOffset()은 종료된다.

두 클래스 모두 drawHere() 메서드는 화살표에서 꼬리보다 위를 그리는 drawTop(), 꼬리가 있는 줄을 그리는 drawTail(), 꼬리보다 아래를 그리는 drawBottom() 메서드를 호출한다.

RightArrow의 drawTop() 메서드는 가장 위에 점부터 그려나가야 하므로 시작하는 지점을 나타내는 변수 startOfLine을 offset + tail – 1로 저장한다. 꼬리보다 윗부분을 그리므로 lineCount = width/2 -1 로 한다. \*과 \*사이의 간격인 insideWidth은 처음에는 1로 저장해둔다. 이후 lineCount만큼 반복문을 실행하며 insideWidth를 매번 2씩 더해 \*과 \*사이의 간격을 늘려준다. drawTail()메서드는 offset만큼 칸을 띄어준 뒤, tail의 길이만큼 \*을 출력한다. 이후 화살표의 머리 부분의 점을 그려야하므로 꼬리와 점 사이 간격 insideWidth를 2\*(width/2-1)로 저장한 후, 그 간격만큼 띄어준 뒤 \*를 출력한다. drawBottom()은 drawTop()과는 달리 반복문을 실행할 때 마다 \*과 \*사이의 간격이 줄어들어야 하므로 먼저 처음의 간격 insideWidth를 1 + 2\*(width/2 -1 -1)로 저장해준다. 이후 반복문을 실행해나가며 간격 insideWidth를 2만큼 줄여나간다. LeftArrow의 drawTop(), drawTail(), drwBottom()의 기본적인 구조는 RightArrow의 메서드와 유사하나 그리기 시작하는 지점과 \*과 \*사이의 간격을 유의하며 메서드를 작성한다.