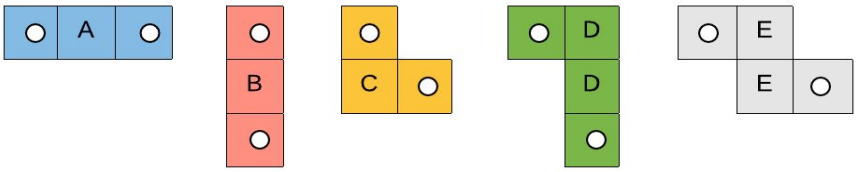
2. ACSL Stretch

PROBLEM: Given a rectangular grid and the 5 types of pieces shown below, the object of Stretch is to place pieces in the grid so that they form a connected path across the grid. If the initial piece is in the top row, the path goes to the bottom row. If the initial piece is in the bottom row, the path goes to the top row.



- A piece cannot be rotated or flipped.
- A piece can only connect to the last piece placed.
- Pieces connect only at circle tiles and the tiles with the circles are the only tiles that are allowed to touch.
- A piece cannot be placed in the grid such that it would cover any part of another piece, cover a blocked cell, or extend beyond the grid.
- The one and only one tile allowed to touch the top or bottom boundary side is a circle tile. No tile can touch the left or right boundary side.
- Pieces are placed in alphabetical order. If a piece does not fit, skip it and use the next piece that fits. When Piece E is either used or skipped, then begin again with Piece A.
- Grid cells are numbered consecutively starting with 1 in the upper left corner and continue from left to right and from top to bottom.
- We guarantee that if a piece can be placed using the rules above, it will be the only piece that will fit.

EXAMPLE: The example above is a 6 x 10 grid with a starting cell of 4. There are blocked cells at cells 16, 23 and 34. Piece A can not be placed at location 4. Piece B is placed at location 4. Piece C can only be added at location 25. Piece D can only be added at location 37 and it touches the opposite side and ends the path. The path is BCD.

INPUT: There will be 10 lines of data. Each line will contain the numbers r, c, s, n, followed by n numbers. r indicates the number of rows in the grid. c indicates the number of columns in the grid. s

2. ACSL Stretch

indicates the starting cell number for the first piece. *n* indicates the number of blocked cells. The next *n* numbers are the cells that are designated as blocked. Numbers are separated by one or more spaces.

OUTPUT: Form a path from the starting cell to the opposite edge of the grid using the algorithm above. Print the sequence of pieces that were used to form the path in the order that they were placed in the grid. We guarantee there will always be a unique path.

SAMPLE INPUT ***(****3 lines of data only; the Test Data will have 10 lines of data****):***

6 10 4 3 16 23 34
12 12 6 5 31 56 104 29 91
12 12 138 6 26 28 62 89 102 99

SAMPLE OUTPUT:

1. **BCD**
2. **BCDEB**
3. **BCDBC**

2. ACSL Stretch

TEST DATA

TEST INPUT:

```
9 8 2 4 18 19 30 36
8 9 2 5 23 40 50 60 29
12 12 2 10 38 41 78 107 115 116 76 88 92 94
12 12 10 13 20 35 46 43 6 57 66 53 69 117 91 101 116
12 12 6 10 42 29 45 66 93 90 65 104 100 113
12 12 138 5 102 88 98 39 40
12 12 135 7 100 62 64 78 89 41 26
12 12 143 7 107 93 67 81 38 62 103
12 12 142 10 117 107 81 91 66 28 69 29 50 52
12 12 139 14 46 114 103 104 141 95 53 69 54 62 75 77 15 31
```

TEST OUTPUT:

1. **CDEB**
2. **BCDB**
3. **BCDEAB**
4. **BABCDAB**
5. **BCDABCD**
6. **BCDBAB**
7. **BCABCAB**
8. **BCDEABC**
9. **BCDEAB**
10. **BEBDEABEAB**