Plant a Christmas Tree

Evergreen trees are really wonderful. They were treasured by all civilisations of the Western world, from ancient Egiptian priests to Celtic druids in the British Isles. In the late Middle Ages a tradition of placing an evergreen tree at home for Christmas developed in Germany and spread throughout the Old and New World, reaching the Asia Pacific and America in the XIX-th century.

Surely, you must have noticed the sad fact that nowadays this global custom, however beautiful it may be, results in the death of millions of coniferous trees worldwide. Help us in our effort to restore the healthy balance. In the Christmas period, draw & plant your own tiny fir tree!

Since we are limited to text mode, there is little room for creative art, and solid, well built trees are definitely favoured. An *ideal tree* consists of several lines (at least 1) of the same length, consisiting of ASCII characters -- both whitespace ("spaces"), and non-whitespace ("relevant characters"). Counting from the top, the number of characters between the first and last relevant characters in a line (inclusive) is equal to 1, 1, 3, 1, 3, 5, 1, 3, 5, 7, 1, 3,... for consecutive lines. The line for which this distance is the largest begins and ends with relevant characters. All other lines contain exactly the same number of spaces to the left of the leftmost relevant character and to the right of the rightmost relevant character (this gives the ideal tree a nice, vertical trunk).

Please write a program which outputs a tree as close to an ideal tree as you can get, and keeps it as small as possible (such a tree has the largest chance of sprouting roots when planted). *And it can hardly come as a surprise to you to learn that the source code of the program you submit has to be identical to the text it writes to output (character by character, there are no exceptions)!*

Score

Your program will be judged as follows: if the program is not a quine (i.e. if it contains no relevant characters or outputs text different than its own source code) it will be judged as a Wrong Answer. Any other program will receive some number of penalty points depending on its size and quality as a tree (the fewer points, the better). One penalty point is given for every line of code used. 10 penalty points are given for a line without any relevant characters (how can you expect a broken tree to grow?). For non-empty lines, the position of the leftmost and rightmost relevant characters in the analyzed tree are compared with respect to corresponding positions in an ideal tree with the same number of lines. The squared differences in position between these two pairs of characters are added to the penalty score.

Technical note: a single newline character (ASCII 10) should be used to terminate all lines. ASCII characters 32 (space) and 9 (tab) are treated as single spaces, all other characters are considered relevant. Notice that the problem description doesn't penalise for left out or excessive spaces after the last relevant character of a line (but doesn't allow any difference between the source code and output text in this respect).

Example

C source code:

```
  ;
  ;
 ;;;
```

```
   ;
  main(
 )  {;
    ;
  ; ;
 /* */
;return
   0
  ;;}
```

This code would be judged as Wrong Answer, since it isn't a valid quine. Were it a quine, it would receive 16 penalty points (12 for 12 lines, $4=2^2$ additional penalty points for a misplaced rightmost relevant character in line 5).

Solutions to this problem may only be submitted in the following languages: C, C++, Pascal, Java, C#, Python, Haskell, OCaml, Brainfk, Intercal.**