Parser Rule:

```
line       : build EOF;
LEFT_PAREN  : '(';
RIGHT_PAREN : ')';
build      : (LEFT_PAREN build RIGHT_PAREN)*;
```

Informal proof of correctness:

It suffices to prove the following two things:

1) Every valid balanced parentheses (BP) can be described by the recursive definition of build.
2) Any string created from build must be a BP.

We proceed inductively, by proving (1) and (2) on BP's of length 2k, for all non-negative integers k. It is trivial to prove that BP's cannot have odd length.

Base case: k = 0. 1) clearly holds because the length 0 BP is the empty string, and since any grammar alpha* implies 0 or more alphas, (1) holds here. Similarly, the only length-0 string that can be created from build is the empty string, which is a BP.

We can also easily establish (1) and (2) for base cases k=1 and k=2.

Inductive step: Suppose (1) and (2) hold for length-*2n* cases, *n*>=2. We will establish that they hold for length-2(*n+1*) string, *S*. The proof follows easily from observing construction of build. *S* must be a concatenation of more than 1 smaller builds (in which case, *S* is valid, for a concatenation of BP's necessarily results in a BP) or *S* is of the format *left_paren build right_paren.* Our inductive assumption assures us that *build* is valid, hence *S* must be valid as well. This proves (2).

(1) follows from noting that any BP must be a concatenation of smaller BP's, or must otherwise be a smaller BP wrapped in left and right parentheses. This is precisely the definition of build.