You have 60 minutes.

Show your work and explain your answers. Good work often earns partial credit. A correct answer with no explanation often earns little or no credit.

If you are asked to write code but you do not know the exact Python required, then try to write code that is approximately correct. If you think that your code does not demonstrate that you understand the solution, then describe your idea in English as well. Be precise enough that I cannot misinterpret your solution.

Good luck.

In the AVL trees below, keys are shown but values are not. Following our standard algorithm, insert the key `5` into the following AVL tree and restore balance.

Following our standard algorithm, delete the key `8` from this AVL tree and restore balance.

Write a function `parse` that parses postfix algebraic expressions such as `3 x + 15 *`. You may assume that there are no parentheses in the input, that the user makes no syntax errors, and that all operators are binary. You may assume the existence of a function `isOperator`, that for any token returns `True` or `False` indicating whether the token is an operator.

Describe the running time of `__setitem__` in `Dictionary` implemented atop a binary search tree (with no balancing). You may want to comment on several possible cases.

Describe the running time of `__setitem__` in `Dictionary` implemented atop a AVL-balanced binary search tree. You may want to comment on several possible cases.

This question has two ingredients. First, our textbook's parsing algorithm stores operands as values (e.g. the integer `3`) rather than as tokens (e.g. the string `"3"`) in the parse tree. Second, the epilogue of the Interpretation assignment talks about user-defined functions stored in the environment as parse trees. Now suppose that we're trying to add user-defined functions to our interpreter, and we have a user who has volunteered to help us test the interpreter as we work on it. Somewhere in the middle of using our interpreter, the user issues the following command.

```
>> myFunction = (x return (x + y))
```

Should `x` be stored in the parse tree as a token or as a value? Should `y` be stored in the parse tree as a token or as a value? Discuss.