Category

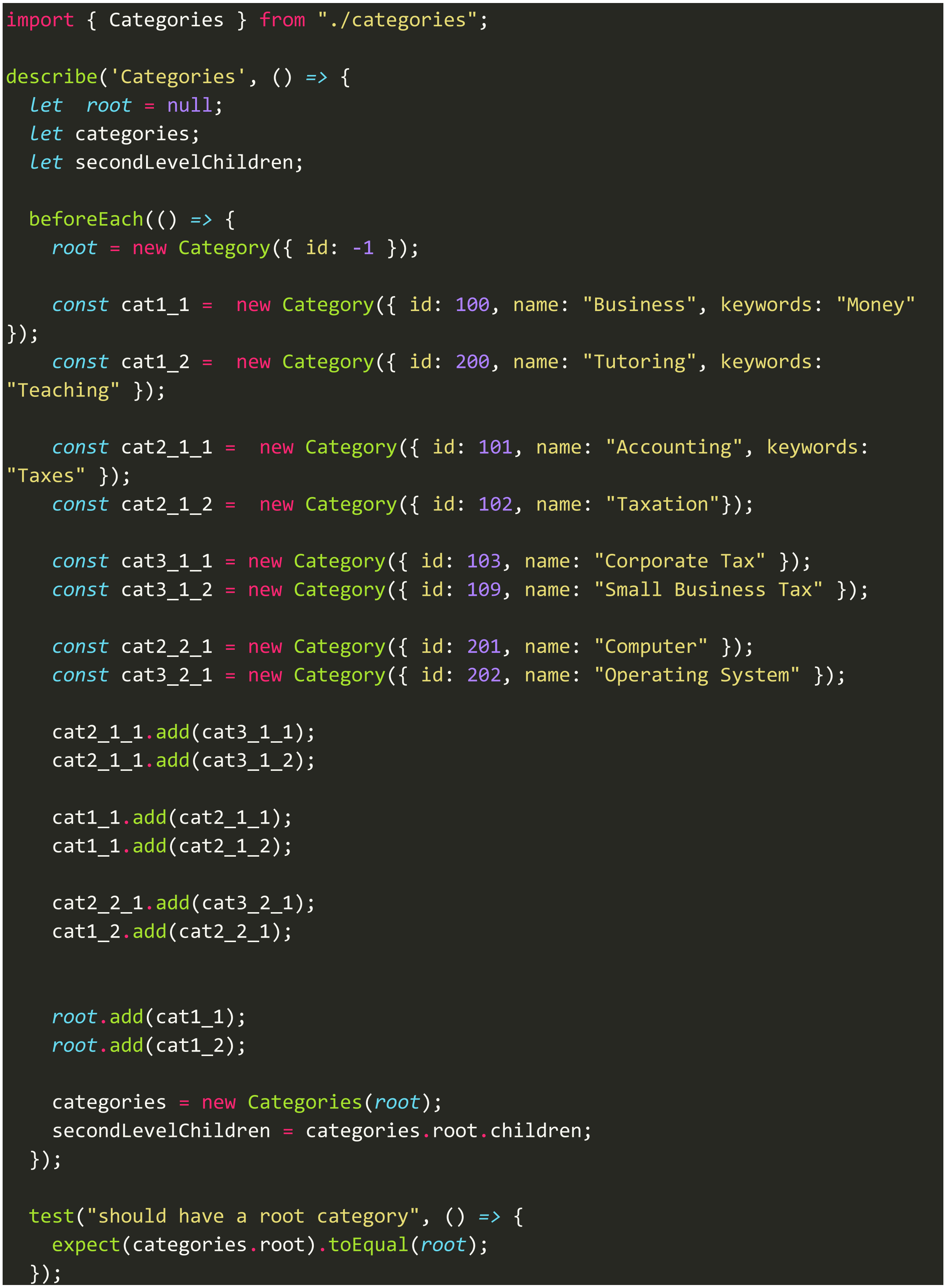
```
class Category {
  constructor({ id, name = '', keywords = '' }) {
    this.id = id;
    this.name = name;
    this.keywords = keywords;
    this.parentCategoryId = null;
    this.parent = null;
    this.children = [];
  }

  add(node) {
    node.parent = this;
    node.parentCategoryId = this.id;
    this.children.push(node);
  }
}

export {Category};
```

Categories

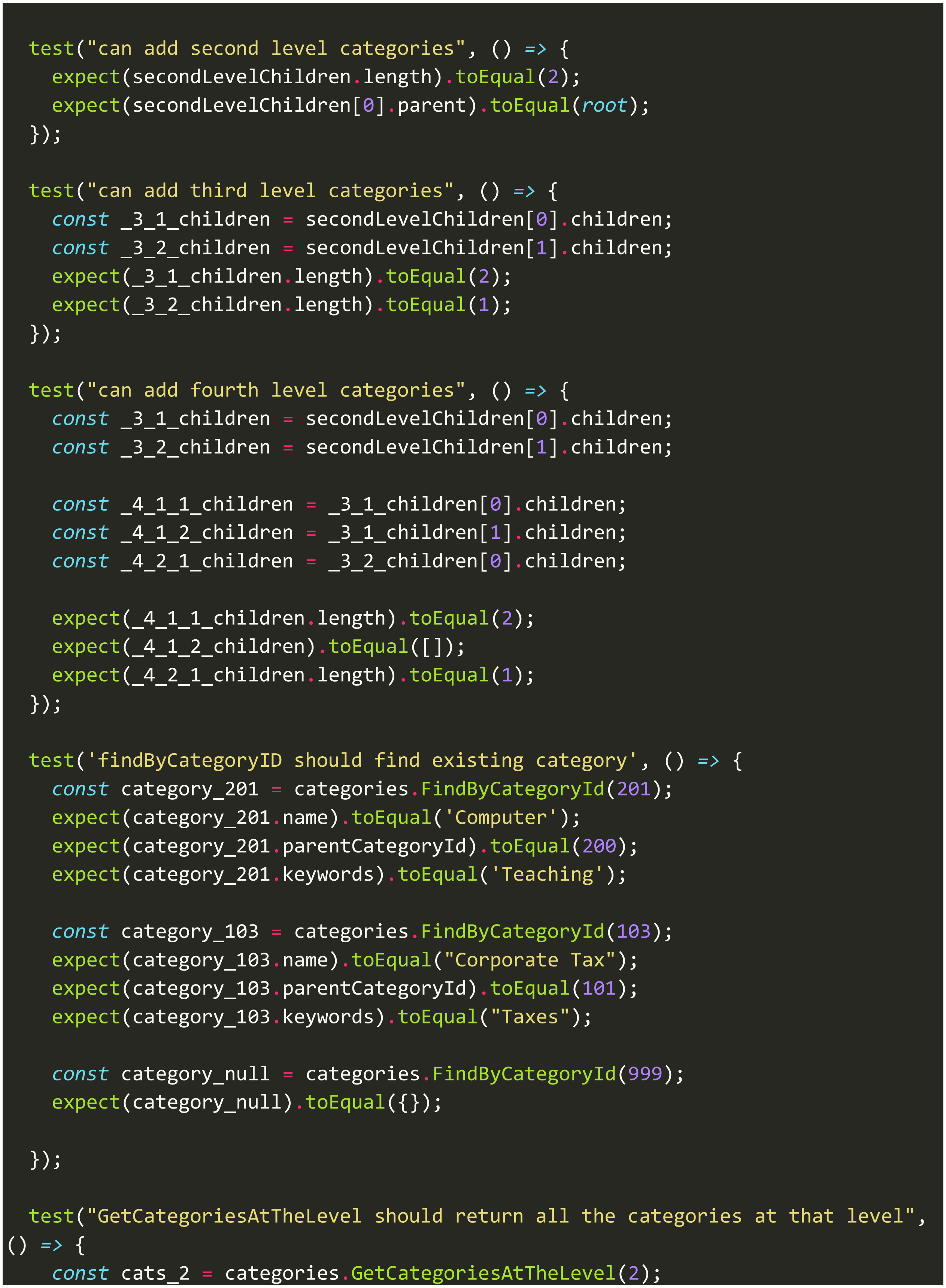
```
class Categories {
  constructor(rootNode) {
    this.root= rootNode;
  }

  FindByCategoryId(id) {
    //why breadth first traverse
    //
    let retObj = {};
    const nodes= [this.root];

    while(nodes.length) {
      const node = nodes.shift();
      if(node.id == id) {
        retObj = node;
        retObj.keywords = this._getKeywords(node);
        break;
      } else {
        nodes.push(...node.children)
      }
    }
    return retObj;
  }

  GetCategoriesAtTheLevel(tree_level) {
    let node_level = 0;
    const retCategories = [];
    const nodes = [this.root, 'LEVEL_UP'];
    while (nodes.length) {
      const node = nodes.shift();
      if (node === "LEVEL_UP") {
        if (node_level == tree_level) {
          break;
        } else {
          node_level++;
          nodes.push(node);
        }
      } else {
        if (node_level == tree_level) {
          retCategories.push(node.id);
        }
        nodes.push(...node.children);

      }
    }
    return retCategories;
  }

  // private function but still it will be exposed outside.
  _getKeywords(node) {
    if (node.keywords && node.keywords.length > 0) {
      return node.keywords;
    } else {
      return this._getKeywords(node.parent);
    }
  }
}

export {Categories};
```

TESTS

```
import {Category} from './category';
```



```
import { Categories } from "./categories";

describe('Categories', () => {
  let  root = null;
  let categories;
  let secondLevelChildren;

  beforeEach(() => {
    root = new Category({ id: -1 });

    const cat1_1 =  new Category({ id: 100, name: "Business", keywords: "Money"
});
    const cat1_2 =  new Category({ id: 200, name: "Tutoring", keywords:
"Teaching" });

    const cat2_1_1 =  new Category({ id: 101, name: "Accounting", keywords:
"Taxes" });
    const cat2_1_2 =  new Category({ id: 102, name: "Taxation"});

    const cat3_1_1 = new Category({ id: 103, name: "Corporate Tax" });
    const cat3_1_2 = new Category({ id: 109, name: "Small Business Tax" });

    const cat2_2_1 = new Category({ id: 201, name: "Computer" });
    const cat3_2_1 = new Category({ id: 202, name: "Operating System" });

    cat2_1_1.add(cat3_1_1);
    cat2_1_1.add(cat3_1_2);

    cat1_1.add(cat2_1_1);
    cat1_1.add(cat2_1_2);

    cat2_2_1.add(cat3_2_1);
    cat1_2.add(cat2_2_1);


    root.add(cat1_1);
    root.add(cat1_2);

    categories = new Categories(root);
    secondLevelChildren = categories.root.children;
  });

  test("should have a root category", () => {
    expect(categories.root).toEqual(root);
  });
```



```
  test("can add second level categories", () => {
    expect(secondLevelChildren.length).toEqual(2);
    expect(secondLevelChildren[0].parent).toEqual(root);
  });

  test("can add third level categories", () => {
    const _3_1_children = secondLevelChildren[0].children;
    const _3_2_children = secondLevelChildren[1].children;
    expect(_3_1_children.length).toEqual(2);
    expect(_3_2_children.length).toEqual(1);
  });

  test("can add fourth level categories", () => {
    const _3_1_children = secondLevelChildren[0].children;
    const _3_2_children = secondLevelChildren[1].children;

    const _4_1_1_children = _3_1_children[0].children;
    const _4_1_2_children = _3_1_children[1].children;
    const _4_2_1_children = _3_2_children[0].children;

    expect(_4_1_1_children.length).toEqual(2);
    expect(_4_1_2_children).toEqual([]);
    expect(_4_2_1_children.length).toEqual(1);
  });

  test('findByCategoryID should find existing category', () => {
    const category_201 = categories.FindByCategoryId(201);
    expect(category_201.name).toEqual('Computer');
    expect(category_201.parentCategoryId).toEqual(200);
    expect(category_201.keywords).toEqual('Teaching');

    const category_103 = categories.FindByCategoryId(103);
    expect(category_103.name).toEqual("Corporate Tax");
    expect(category_103.parentCategoryId).toEqual(101);
    expect(category_103.keywords).toEqual("Taxes");

    const category_null = categories.FindByCategoryId(999);
    expect(category_null).toEqual({});

  });

  test("GetCategoriesAtTheLevel should return all the categories at that level",
() => {
    const cats_2 = categories.GetCategoriesAtTheLevel(2);
    expect(cats_2.length).toEqual(3);
    expect(cats_2).toEqual([101,102,201]);
    const cats_3 = categories.GetCategoriesAtTheLevel(3);
    expect(cats_3.length).toEqual(3);
    expect(cats_3).toEqual([103, 109, 202]);
    const cats_level_not_exist = categories.GetCategoriesAtTheLevel(6);
    expect(cats_level_not_exist.length).toEqual(0);
  });
});
```