Testing Your Interpreter, Part 4

Here are some tests for your part 4 interpreter.

Basic Tests

Test 1: This code should return 10.

```
class A {
  static function main() {
    return 10;
  }
}
```

Test 2: This code should return true.

```
class A {
  static var x = true;
  static function main() {
    return x;
  }
}
```

Test 3: This code should return 30.

```
class A {
  static var x = 30;
  static function main() {
    return A.x;
  }
}
```

Test 4: This code should return false.

```
class A {
  static var x = true;
  static var y = false;

  static function and(a, b) {
    return a && b;
  }

  static function main() {
    return A.and(A.x, A.y);
  }
}
```

Test 5: This code should return 30 when running A's main, but 510 when running B's main.

```
class A {
  static var x = 10;
  static var y = 20;

  static function add(a, b) {
    return a + b;
```

```
  }

  static function main() {
    return A.add(A.x, y);
  }
}

class B extends A {
  static var y = 200;
  static var z = 300;

  static function main() {
    return add(x+y,z);
  }
}
```

Test 6: This code should return 30 when running A's main and 530 when running B's main.

```
class A {
  static var x = 10;
  static var y = 20;

  static function add(a, b) {
    return a + b;
  }

  static function main() {
    return A.add(x, y);
  }
}

class B extends A {
  static var y = 200;
  static var z = 300;

  static function main() {
    return add(x+y,super.y + z);
  }
}
```

Test 7: This code should return 105 when running A's main and 1155 when running B's main.

```
class A {
  static var x = 49*5*9;
  static var y = 7*25*3;

  static function gcd(a,b) {
    if (a < b) {
      var temp = a;
      a = b;
      b = temp;
    }
    var r = a % b;
    while (r != 0) {
      a = b;
      b = r;
      r = a % b;
    }
    return b;
  }
```

```
  static function main() {
    return gcd(x, y);
  }
}

class B extends A {
  static var y = super.y * 121;
  static var x = super.x * 11;

  static function main() {
    return gcd(x,y);
  }
}
```

Test 8: This code should return 615.

```
class A {
  static var a = 1;
  static var b = 10;

  static function setA(x) {
    a = x;
  }

  static function getSum() {
    return a + b;
  }
}

class B {
  static function main() {
    A.setA(5);

    return A.getSum() + C.x + C.timesX(A.a);
  }
}

class C {
  static var x = 100;
  static function timesX(a) {
    return a * x;
  }
}
```

Test 9: This code should give an error when running B's main and return 4321 when running C's main.

```
class A {
  static var a = 1;
  static var b = 20;
}

class B extends A {
  static var c = 300;

  static function main() {
    return a + b + c + d;
  }
}
```

```
class C extends B {
  static var d = 4000;

  static function main() {
    return a + b + c + d;
  }
}
```

Test 10: This code should return 400 when running Square's main.

```
class Rectangle {
  static var width = 10;
  static var height = 12;

  static function area() {
    var a = width * height;
    return a;
  }

  static function setSize(x, y) {
    width = x;
    height = y;
  }
}

class Square extends Rectangle {
  static function setSize(x) {
    super.setSize(x, x);
  }

  static function main() {
    setSize(20);
    return area();
  }
}
```

Test 11: This code should return 15.

```
class A {
  static var total = 0;

  static function sum(x) {
    if (x > 0) {
      total = total + x;
      sum(x - 1);
    }
  }

  static function main() {
    sum(5);
    return total;
  }
}
```

Test 12 should return 125 when running A's main.

```
class A {
  static function divide(x, y) {
```

```
    if (y == 0)
      throw y;
    return x / y;
  }

  static function main() {
    var x;

    try {
      x = divide(10, 5) * 10;
      x = x + divide(5, 1);
    }
    catch(e) {
      x = e;
    }
    finally {
      x = x + 100;
    }
    return x;
  }
}
```

Test 13 should return 100 when running A's main.

```
class A {
  static function divide(x, y) {
    if (y == 0)
      throw y;
    return x / y;
  }

  static function main() {
    var x;

    try {
      x = divide(10, 5) * 10;
      x = x + divide(5, 0);
    }
    catch(e) {
      x = e;
    }
    finally {
      x = x + 100;
    }
    return x;
  }
}
```

Test 14 should return 2000400 when running A's main.

```
class A {
  static function divide(x, y) {
    if (y == 0)
      throw y;
    return x / y;
  }

  static function main() {
    var x = 0;
```

```
    var j = 1;

    try {
     while (j >= 0) {
      var i = 10;
      while (i >= 0) {
        try {
          x = x + divide(10*i, i);
        }
        catch(e) {
          x = x + divide(e, j);
        }
        i = i - 1;
      }
      j = j - 1;
     }
    }
    catch (e2) {
      x = x * 2;
    }
    return x;
  }
}
```

Test 15: Functions inside functions inside functions. This code should return 64 running Pow's main.

```
class Pow {
  static function main() {
    var result;
    var base;

    function getpow(a) {
       var x;

       function setanswer(n) {
          result = n;
       }

       function recurse(m) {
         if (m > 0) {
           x = x * base;
           recurse(m-1);
         }
         else
           setanswer(x);
       }

       x = 1;
       recurse(a);
    }
    base = 2;
    getpow(6);
    return result;
  }
}
```

Additional Tests for Method Overloading

Test 16: This code should return 530.

```
class A {
  static var x = 10;
  static var y = 20;

  static function add(a, b) {
    return a + b;
  }

  static function add(a,b,c) {
    return a + b + c;
  }

  static function main() {
    return A.add(x, y) + A.add(x, y, y) * 10;
  }
}
```

Test 17: This code should return 66.

```
class A {
  static var x = 10;
  static var y = 20;

  static function add(a, b) {
    return a + b;
  }

  static function add(a,b,c) {
    return a + b + c;
  }
}

class B extends A {
  static var x = 2;
  static var y = 30;

  static function add(a,b) {
    return a*b;
  }

  static function main() {
    return add(x,y) + add(x,x,x);
  }
}
```