Ice Cream Inheritance

You are the owner of an ice cream parlor. With so many places now passing laws that restaurants have to post calorie counts for their products, you decide to write a program to help you calculate these numbers.

(1) Create a class to represent an ice cream sundae. This class should contain:

- One **private** field, representing the number of scoops in a sundae.
- One public constructor, taking one integer argument that represents the number of scoops desired in this sundae.
- One public **const** method, taking no arguments, which returns an int representing the number of calories in the sundae. Google says one scoop of ice cream has 137 calories.

Example of using the class:

```
sundae yummy(2);                          // create a 2-scoop sundae
cout << yummy.get_calories() << endl;     // prints 274 calories
```

(2) Create a class to represent a banana split. This class should inherit from your ice cream sundae class. A banana split is simply an ice cream sundae that also contains a number of sliced bananas. This class should contain:

- One **private** field, representing the number of bananas in the sundae.
- One public constructor, taking two integer arguments that represent the number of desired ice cream scoops and bananas, respectively, in the sundae.
    - Hint: to implement this, you will need to have the banana split constructor explicitly call the sundae constructor, because you cannot directly set the number of scoops (it's private).
- One public **const** method, overriding the method in sundae, to return the number of calories in the banana split. Google says one small banana has 90 calories.
    - Hint: to implement this, you will need to have the banana split get_calories method call the sundae get_calories method, because you cannot access the number of scoops of ice cream from the banana split class.

Example of using the class:

```
bsplit tasty(3, 1);                       // create a 3-scoop, 1-banana sundae
cout << tasty.get_calories() << endl;     // prints 501 calories
```

(3) Challenge: Override operator<< (twice) to allow sundaes and banana splits to be printed:

```
cout << yummy << tasty << endl;
// might print something like:
      This sundae has 274 calories.
      This banana split has 501 calories.
```

Override operator+ to allow sundaes to be added together:

```
sundae gonna_be_sick = yummy + yummy + yummy; // 6-scoop sundae
```