Mistake: -1

```
-Case 1:
+Case1:
```

Please make sure to match your input/output with our input/output specifications.

Mistake: -5

```
double maxValue = 3.4e38;
double minValue = -3.4e38;
```

You cannot hard-code the maximum and minimum single-precisison floating point numbers as 3.4e38 and -3.4e38. These values are not correct. Use `std::numeric_limits<float>`.

Mistake: -5

```
original_health + input > max_health
```

Your check for overlow is not correct. The addition of the current boss's health and the input will in principle result in an overflow before it is compared to the maximum short value. The overflow did not actually happen here because C++ implicitly casts all types smaller than int to int before doing arithmetic, but in principle this check does not work.

Mistake: -5

```
if (attack > 0 && newhealth < 0) {
```

Your did not correctly prevent the overflow from happening. Here it happens and then gets corrected, but we want to prevent the overflow from happening in the first place.

Mistake: -0

Using atof/atoi/atod to parse numbers from strings

Mistake: -0

`heal_attack > 767`

Do not hard-code the value 767 in your program. You should use std::numeric_limit to find the maximum short number and subtract it from your current health to detect a potential overflow.

Mistake: -2

```
if (isGoodFloat < std::numeric_limits<float>::max() &&
    isGoodFloat > std::numeric_limits<float>::lowest())
```

The check for when the number will not overflow a float is not exactly correct. Instead of using < and > you should use <= and >=.

Mistake: -0

```
for(int i = 0; i < testCases; i++)
{
    std::cin >> tempString;
    myStrings.push_back(tempString);
}
```

You do not have to (and should not) store all the inputs in a vector before processing them. It's cleaner to just process inputs one by one as they come. "'

Score Summary:

- Mean = 84.21232876712328
- Std dev = 19.024275027838915
- Median = 92.0
- Max = 100.0
- Min = 8.0

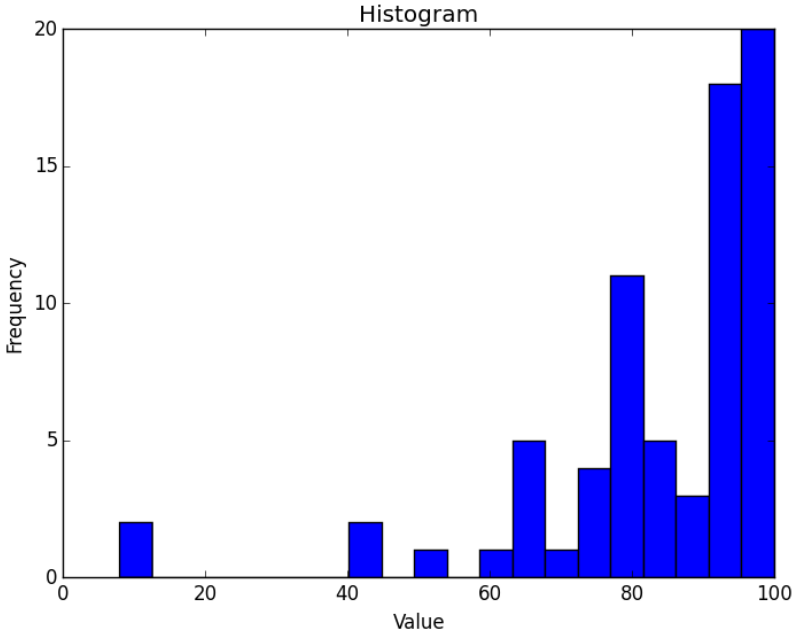

Figure 1: Homework 1 Score Distribution