

19<sup>th</sup> Annual  
South Jersey Regional  
High School Programming Contest

hosted by the  
Rowan University Computer Science Department

Saturday, 9 April 2005

Contest Problem

# 1 Background

Large-scale manufacturing requires automated equipment with many adjustments. Setting these adjustments, and otherwise controlling the manufacturing process, is called *process control*.

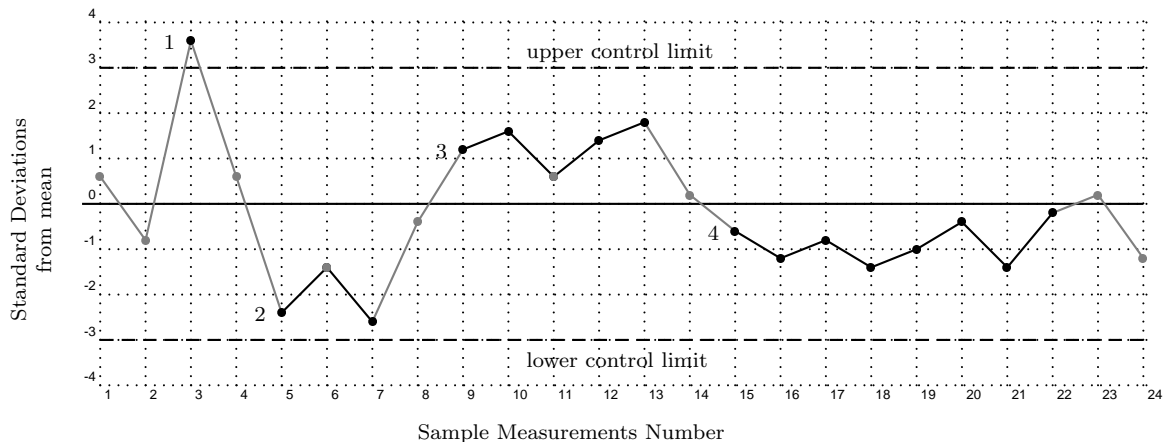
For example, potato chips will be taste-tested with varying amounts of salt on them, to determine what level of saltiness tastes best to the greatest number of people. The company will set their machines to salt the chips accordingly. Too much, or too little, salt means the batch won't taste right, and has to be discarded. Since incorrect adjustments cut into profit margins, process control is an important issue. Modern production facilities use *Statistical Process Control*, which is a way of using statistical principles to improve operations.

Statistical process control takes into account that some variation between production samples is to be expected, and determines mathematically how much variation is acceptable. First, a number of batches are made and taste-tested. The batches which pass the taste test are then explicitly measured for saltiness. These measurements are used to find the average amount of salt in acceptable batches, known as the *mean*, and also to find how wide the range of acceptable salt levels is, based on the *standard deviation*. Once the mean and standard deviation are known, they can be used to control the adjustment of the equipment.

One method of statistical process control is as follows: several samples of chips are taken from the production line at regular intervals. The salt percentage for each sample is measured, and their average is then plotted against the known mean and standard deviation. The salt machine must be adjusted if any of the following conditions (known as 'decision rules') is met:

1. The measured salt content is off the target mean by more than 3 times the standard deviation (said to be 'out of control').
2. In three consecutive sample averages, there are two which are off by more than two standard deviations on the same side.
3. In five consecutive sample averages, there are four which are off by more than one standard deviation on the same side.
4. The salt content is on the same side of the target mean for eight sample averages in a row.

The following chart illustrates these four conditions:



(This chart corresponds to the first data set on page 5.)

Your task in this programming contest is to read the measured values of saltiness in potato chip bags which have passed quality control and taste tests, and find the mean and standard deviation for those bags. Then, read in values for more batches, checking those against your computed mean and standard deviation. Your program will indicate whether, and which way, the salt machine needs to be adjusted.

## 2 Statistical Computations

### 2.1 Mean

To compute the mean of a set of sample data, you add all the values in set and divide by the number of elements. The mean of the set  $\{2.3, 8.7, 1.6, 12.5, 16.2\}$  is:

$$\bar{x} = \frac{(2.3 + 8.7 + 1.6 + 12.5 + 16.2)}{5} = \frac{41.3}{5} = 8.26$$

Note that you do not need to store all the numbers, merely maintain a running total and a count.

### 2.2 Standard Deviation

The formula for the standard deviation is more complex. One mathematical notation is:

$$s = \sqrt{\frac{\sum_{i=1}^n (x_i^2) - \frac{(\sum_{i=1}^n x_i)^2}{n}}{n - 1}}$$

That is, the sum of the squares of the numbers, minus the square of the sum of the numbers over the number of numbers, then divided by one less than the number of numbers. The square root of that value is the standard deviation. For the samples  $\{2, 3, 4\}$ , this gives:

$$\begin{aligned} s &= \sqrt{\frac{(2^2 + 3^2 + 4^2) - \frac{(2+3+4)^2}{3}}{3 - 1}} \\ &= \sqrt{\frac{29 - \frac{81}{3}}{2}} \\ &= \sqrt{\frac{29 - 27}{2}} \\ &= \sqrt{\frac{2}{2}} \\ &= \sqrt{1} \\ &= 1 \end{aligned}$$

Note again that you do not need to store all the numbers: you need only to maintain a running sum (as for the mean), a running sum of each number squared, and a count.

For the samples in section 2.1, whose mean is 8.26, the standard deviation is:

$$\begin{aligned} s &= \sqrt{\frac{(2.3^2 + 8.7^2 + 1.6^2 + 12.5^2 + 16.2^2) - \frac{(2.3+8.7+1.6+12.5+16.2)^2}{5}}{5 - 1}} \\ &= \sqrt{\frac{502.23 - \frac{41.3^2}{5}}{4}} \\ &= \sqrt{\frac{502.23 - 341.138}{4}} \\ &= \sqrt{40.273} \\ &= 6.3461 \end{aligned}$$

### 3 Input/Output Specification

#### 3.1 Input

For text input, your program should accept input in the following format:

1. A single integer,  $\mathcal{D}$ , where  $\mathcal{D} \geq 1$ , which specifies the number of data sets to be scanned.
2.  $\mathcal{D}$  data sets, each of which is in this format:
  - (a) A single integer,  $\mathcal{S}$ , where  $1 \leq \mathcal{S} \leq 100$ , which specifies the number of known-good samples whose mean and standard deviation you are to compute.
  - (b)  $\mathcal{S}$  numbers, one per line, which are the values for the known-good samples. As these values are percentages, they will be between 0.0 and 100.0, inclusive.
  - (c) A single integer,  $\mathcal{P}$ , where  $0 \leq \mathcal{P} \leq 100$ , which specifies the number of samples taken from the production line that will need to be tested.
  - (d)  $\mathcal{P}$  numbers, one per line, which are the measured values from those samples. As these values are percentages, they will be between 0.0 and 100.0, inclusive.

You need not do error-checking on the input. There will be nothing on the first line except a single integer. For each data set, there will be nothing on the first line except a single integer, and nothing on the following lines but the a single value per line. After the known-good sample data, there will be only a single integer on the next line, and the following lines will each have exactly one value.

You may choose to have your program read the input from the keyboard, or ask the user for a filename and then read the file. Users of GUI-based programming environments may prefer to use text boxes into which the values can be entered, and buttons to begin their calculation. Any reasonable variation in the spirit of the problem is acceptable.

#### 3.2 Output

For each data set, your program must generate output according to the following description:

1. The text **Data set  $D$ :**, indicating which data set is being reported on.
2. A line with the text **Control samples:  $S$** , indicating how many good samples were present.
3. A line with the text **Control mean:  $M$** , indicating the arithmetic mean of the good samples.
4. A line with the text **Control std dev:  $D$** , indicating the standard deviation of the good samples.
5. A line with the text **Production samples:  $P$** , indicating how many production samples are present.
6. For each production sample, a line of the form: ' **$N$ : *value*, *distance***' where  $N$  is the sample number, *value* is the measured value of the sample, and *distance* is how much this value is off from the mean in standard deviations.

If this value causes a sequence of values to match one of the decision rules described in section 1, indicate that adjustment is necessary and why. (Note: if the machines are adjusted, you should re-set any counters which are keeping track of runs in the data, but **do not** reset the sample counter.)

Your output does **not** have to duplicate the sample output as regards spacing or use of upper/lower case, or number of decimal places. Your output should be neat, but need not exactly match the sample.

## 4 Sample Data

### 4.1 Sample Input #1

```

2      this file has 2 data sets
10     data set 1 has 10 control samples
10.0  }
9.6   }
9.9   }
8.35  }
11.49 } the measured values of the
9.5   } control samples
11.5  }
9.01  }
10.0  }
10.65 }
24     data set 1 has 24 production samples
10.5  }
9.2   }
13.6  }
10.6  }
7.6   }
8.5   }
7.2   }
9.6   }
11.2  }
11.6  }
10.5  } the measured values of the
11.4  } production samples (note that
11.8  } this corresponds to the chart
10.1  } on page 2)
9.4   }
8.8   }
9.2   }
8.6   }
9.0   }
9.6   }
8.6   }
9.9   }
10.1  }
8.8   }
5      data set 2 has 5 control samples
2.3   }
8.7   } the measured values of the
1.6   } control samples
12.5  }
16.2  }
8      data set 2 has 8 production samples
3.4   }
5.3   }
11.11 }
8.26  } the measured values of the
9.11  } production samples
14.92 }
10.66 }
17.76 }
```

### 4.2 Sample Output #1

```

Data Set 1:
Control samples: 10
Control mean:    10.000
Control std dev: 1.001

Production Samples: 24
1: 10.50, 0.500
2: 9.20, -0.799
3: 13.60, 3.597 lower; rule 1
4: 10.60, 0.599
5: 7.60, -2.398
6: 8.50, -1.499
7: 7.20, -2.798 raise; rule 2
8: 9.60, -0.400
9: 11.20, 1.199
10: 11.60, 1.599
11: 10.50, 0.500
12: 11.40, 1.399
13: 11.80, 1.798 lower; rule 3
14: 10.10, 0.100
15: 9.40, -0.599
16: 8.80, -1.199
17: 9.20, -0.799
18: 8.60, -1.399
19: 9.00, -0.999
20: 9.60, -0.400
21: 8.60, -1.399
22: 9.90, -0.100 raise; rule 4
23: 10.10, 0.100
24: 8.80, -1.199

Data Set 2:
Control samples: 5
Control mean:    8.260
Control std dev: 6.346

Production Samples: 8
1: 3.40, -0.766
2: 5.30, -0.466
3: 11.11, 0.449
4: 8.26, 0.000
5: 9.11, 0.134
6: 14.92, 1.049
7: 10.66, 0.378
8: 17.76, 1.497
```

(Your floppy disk has a copy of this sample input in the file named **sample1.txt**.)

### 4.3 Sample Input #2

```

1      this file has 1 data set
15     data set 1 has 15 control samples
1.66  }
1.52  }
1.58  }
1.72  }
1.68  }
1.57  }
1.58  }
1.62  } the measured values of the
1.68  } control samples
1.57  }
1.58  }
1.73  }
1.62  }
1.43  }
1.64  }
35     data set 1 has 35 production samples
1.43  }
1.56  }
1.39  }
1.62  }
1.74  }
1.73  }
1.59  }
1.80  }
1.75  }
1.60  }
1.59  }
1.63  }
1.64  }
1.36  }
1.65  }
1.64  }
1.85  } the measured values of the
1.66  } production samples
1.61  }
1.53  }
1.60  }
1.59  }
1.55  }
1.56  }
1.47  }
1.59  }
1.58  }
1.75  }
1.53  }
1.62  }
1.60  }
1.36  }
1.70  }
1.85  }
1.67  }
```

### 4.4 Sample Output #2

```

Data Set 1:
Control samples: 15
Control mean:    1.612
Control std dev: 0.078

Production Samples: 35
1:  1.43, -2.319
2:  1.56, -0.663
3:  1.39, -2.829  raise; rule 2
4:  1.62,  0.102
5:  1.74,  1.631
6:  1.73,  1.503
7:  1.59, -0.280
8:  1.80,  2.395
9:  1.75,  1.758  lower; rule 3
10: 1.60, -0.153
11: 1.59, -0.280
12: 1.63,  0.229
13: 1.64,  0.357
14: 1.36, -3.211  raise; rule 1
15: 1.65,  0.484
16: 1.64,  0.357
17: 1.85,  3.032  lower; rule 1
18: 1.66,  0.612
19: 1.61, -0.025
20: 1.53, -1.045
21: 1.60, -0.153
22: 1.59, -0.280
23: 1.55, -0.790
24: 1.56, -0.663
25: 1.47, -1.809
26: 1.59, -0.280  raise; rule 4
27: 1.58, -0.408
28: 1.75,  1.758
29: 1.53, -1.045
30: 1.62,  0.102
31: 1.60, -0.153
32: 1.36, -3.211  raise; rule 1
33: 1.70,  1.121
34: 1.85,  3.032  lower; rule 1
35: 1.67,  0.739
```

(Your floppy disk has a copy of this sample input in the file named **sample2.txt**.)

## 5 Test Data

Run your program on this input and make screenshots showing the output. Your program will also be tested on data known only to the judges.

```
1
12
11.82
11.87
13.43
13.24
11.83
11.95
12.75
13.95
12.89
14.09
12.90
15.28
30
13.09
12.34
12.12
11.42
10.07
12.21
12.03
12.49
12.98
12.61
11.80
10.41
17.85
14.10
13.64
14.57
15.10
15.28
12.82
10.21
15.02
10.17
14.59
15.06
13.86
14.59
15.47
14.88
13.99
14.87
```

(Your floppy disk has a copy of this input in the file named `testdata.txt`.)

## 6 Notes (not required to solve the problem)

Perhaps the most familiar name associated with statistical process control is W. Edwards Deming, who introduced its methods to Japanese companies after the end of World War II. At the end of the war, many economists felt that the Japanese economy would take many decades to rebuild, and until that time the country would be a drain on the United States. Deming, convinced that good management – including, among other things, statistical process control – could get Japanese industry back on its feet, went and got things moving. You can learn more about him at <http://www.deming.org>.

The process control chart, or  $\bar{x}$  chart (pronounced *x-bar chart*), on page 2 is also called a ‘Shewhart Chart’, after Walter A. Shewhart, who is sometimes referred to as ‘the father of statistical quality control’. The American Society for Quality has information about him on their website, at <http://www.asq.org/join/about/history/shewhart.html>.

The video excerpt used to introduce the contest problem is from *Against All Odds: Inside Statistics*, hosted by Teresa M. Amabile. This section was from Program 18, ‘The Sample Mean and Control Charts’. Immediately after the section we watched, there is a profile of W. Edwards Deming, including films of him. You can learn more at <http://www.learner.org/resources/series65.html>. This and other programs run in our area as part of WHYY’s ‘Home College Service’; you can see their broadcast schedules at <http://whyy.org/homecollege/index.html>.

The formula given on page 3 for the standard deviation is derived from this following formula, which is the mathematical definition for standard deviation of a population sample:

$$s = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n - 1}}$$

Which means the square root of (the sum of the squares of the differences between each value and the mean, divided by the number of elements less one).

In other words, for each value, you subtract the mean from that value, and square the result. You add these squares together, and divide that by the number of values less one. Then you take the square root of the result.

While the two formulas are mathematically equal, the one on page 3 is more likely to be subject to roundoff error. The one that requires storing all the numbers, while computationally more complex, is a better choice in situations requiring great precision.

For the samples {2, 3, 4}, you would get a mean of 3, and the standard deviation would be:

$$\begin{aligned} s &= \sqrt{\frac{(2 - 3)^2 + (3 - 3)^2 + (4 - 3)^2}{3 - 1}} \\ &= \sqrt{\frac{(-1)^2 + 0^2 + 1^2}{3 - 1}} \\ &= \sqrt{\frac{1 + 0 + 1}{2}} \\ &= \sqrt{\frac{2}{2}} \\ &= \sqrt{1} \\ &= 1 \end{aligned}$$

Note that this is the same value as computed on page 3.