Program 4 – (0-1 Knapsack)
The program will be to solve the 0-1 knapsack problem using a branch-and-bound strategy. The program will take as input a file containing a number of items, the capacity of the knapsack and the individual items. Each individual item will be on a separate row and consist of a label, then a value then a weight. The label will contain no blank spaces. The output will be a simple message stating the items that should be included in the knapsack and the total weight and value of the knapsack.

Example Data (knapSack1.txt)

```
7
15
A 25 5
B 45 11
C 12 3
AA 7 2
BB 6 2
CC 10 7
AAA 5 4
```

Input
Input file name: **`knapSack1.txt`**

Output:

```
Item included in knapsack: B, AA, BB
Total weight = 15
Total value = 58
```

Specifications:

1) Program driver will be called KnapsackDriver.java
2) The included items must be listed in descending order of their profit to weight ratio.
3) The program will prompt the user for input files.
4) Program must utilize a best-first branch-and-bound strategy.
5) The program is due May 1st and there will be no late program accepted. This is a hard deadline so don't wait till the last day to work on your program.
6) Additional required methods an overloaded method **printItem** this method will take an argument of the either the item number based on the order a descending order of the ratio of the items, or a string containing the item label. The method will print the label, price, and weight and the ratio formatted to 1 decimal point. For example printItem("BB") would print out "BB 6 2 3.0" and printItem(5) would print out "CC 10 7 1.4"