**CS575 Project 4**

**Due at 11:59PM May 5 (Submit through blackboard)**

1. [90%] Implement the following 0/1 knapsack algorithms.
2. [10%] Implement the brute force method to solve the 0/1 knapsack problem. Show the final solution. Specifically, (a) print the total profit and weight; and (b) print the selected items.
3. [80%] Implement the backtracking algorithm for 0/1 knapsack: (a) print the total profit and weight; and (b) print the selected items. Compare the result to the result you have got in 1). If you have implemented the backtracking algorithm correctly, the total profit achieved by this backtracking method must be equal to that achieved by the brute force method implemented in 1).

Implement each algorithm as a separate program. Pass input file, **knapsack.txt,** to each program. Your knapsack.txt should include the number of elements in the first line, weights in the second line, profits in the third line, and knapsack capacity in the fourth line. So, it should look exactly as the following **sample data:**

3

5, 20, 10

50, 140, 60

30

1. [10%] Coding style: Write meaningful comments, while making your code structured, easy to read, and robust.

**Instructions for project submission:**

1. Your code has to work correctly in **remote.cs.binghamton.edu**. You can develop it in environment as you wish but make sure you test it and run it in the retome.cs.binghamton.edu and package it from the same server before submitting. If your code does not work in the mentioned server you will receive no grade points for the respective algorithm. No exception will be considered.
2. Project name should be like this: <language code>\_<userid>\_proj4.tar.gz, all lowercase letters and tar.gz file format.
3. Your source code file names have to be exactly as below: brute.c and backtrack.c . If you are using C++, use cpp extension instead of c.

**Example:** c \_bsmith\_proj4.tar.gz or cpp\_bsmith\_proj4.tar.gz.

When decompressed, a directory **c \_bsmith\_proj4** should be created. Under the directory, there should be **brute.c**, **backtrack.c**, and **Makefile**. Points will be deducted for bad project naming.

Your project should **not** contain any object, executable, readme, input files. It should just include your **C or C++ source code files** and **Makefile**. So, make sure you clean it up before submitting it. Points will be deducted for any presence of unwanted files. While testing, the TA will put knapsack.txt into your project folder to test your projects.

1. The “make” command will only compile all two source code files and it should not run any of the program. After executing “make” command it should create two object files with the following names: brute.out and backtrack.out
2. The brute force algorithm should be executable by **. /brute.out knapsack.txt** and the backtracking algorithm should be executable by **. /backtrack.out knapsack.txt** command. If any of them fails to run using these commands, then expect zero points for the respective algorithm.
3. Please don’t apply your common sense and intuition for packaging, strictly follow the mentioned instructions.
4. Make sure you don’t look into others code and write it for yourself. Copying from Internet sources is strictly not allowed. If you are found doing so, you will receive zero point.
5. If Blackboard says your submission is late, it will be considered late and 10% will be deducted.
6. Contact the TA (szhai2@binghamton.edu) for any clarification needed.  Shuangfei will handle this project.