Change log:

- (version 3) Problem 2: clarified that the code only needs to output the maximum weight of the set, not the subset as a list of elements.
- (version 2) Problem 4: added explicit mention of a maximum limit to the numbers to check for.
- (version 2) Problem 5: simplified probabilistic requirement and fixed runtime goal.
- (version 3) Problem 5: added hint about where to look for probability statement.

This problem set has a total of 5 problems on 2 pages. Written solutions should be submitted to GradeScope before 4:45pm on Wednesday Sep 2.

Your solutions should be submitted according to the guidelines [https://www.cc.gatech.edu/~rpeng/CS4540_F20/ProblemSetGuidelines.pdf](https://www.cc.gatech.edu/~rpeng/CS4540_F20/ProblemSetGuidelines.pdf) In particular:

1. If you choose not to submit a typed write-up, please write neat and legibly.
2. No credit will be given to solutions obtained verbatim from the Internet or other sources, and uploaded codes will be ran through similarity checking software.

Note: try to turn these into optimization versions.

1. Consider the optimization version of the knapsack problem: there are items with size $x_1 \ldots x_n$, and we want to make $Y$ using the fewest number of them, picked with replacement.

Give an algorithm for solving this that takes $O(nY)$ time and $O(Y)$ memory.

Auto judge: [https://dmoj.ca/problem/ccc00s4](https://dmoj.ca/problem/ccc00s4)

2. There are $n$ items, the $i^{th}$ of which has value $1 \leq v_i \leq V$ and positive weight $w_i$. We may select each item at most once to be included in a knapsack of maximum weight capacity $W$. Give an algorithm for finding the maximum sum of values of a subset of items with total weight at most $W$, that runs in $O(n^2V)$ time and takes up $O(nV)$ memory.

Note in particular that the running time of your algorithm cannot depend on the maximum magnitude of the weights.

Auto judge: [https://dmoj.ca/problem/dpe](https://dmoj.ca/problem/dpe)
3. Driving with a gas tank and a credit card.

Consider a grid graph of $1 \leq n \leq N$ rows and $1 \leq m \leq M$ columns: Each cell of the grid contains a vertex of the graph, and each vertex is connected to its immediate neighbours in the north, south, east and west directions by undirected edges. There is a car on the top-left vertex at position (1,1) in the grid. The car has a fuel tank with a maximum capacity of $F > 0$ units and starts with a full tank. It’s destination is the bottom-right vertex at position $(n, m)$.

The car may move to any adjacent vertex in the graph by consuming 1 unit of fuel. If the fuel tank is empty the car cannot move. There are $k$ fuel stations, the $i^{th}$ of which is located in row $1 \leq a_i \leq n$ and column $1 \leq b_i \leq m$. If the car is on a vertex corresponding to the $i^{th}$ fuel station, fuel may be purchased at a price of $c_i > 0$ per unit. Note that the car may not hold more fuel than it’s maximum capacity $F$ at any time, and it may reach its destination with exactly 0 units of fuel remaining.

Give an $O((N + M)NM \log(N + M))$ time, $O(NM(N + M))$ space algorithm to calculate the minimum cost spent on fuel for the car to reach it’s destination, or if it’s not possible to reach the destination due to the fuel tank constraint.

Auto judge: [https://dmoj.ca/problem/cco03p6](https://dmoj.ca/problem/cco03p6)

4. We have 3 positive numbers, $x_1, x_2, x_3$, each at most $X$, and are interested in the numbers that can be written as sums of integer multiples of $x_1, x_2, x_3$. Formally, this is knapsack with three items, with replacement.

Specifically, show that for some arbitrarily limit $L$, we can compute in $O(X \log X)$ time the number of numbers between 1 and $L$ that cannot be written as a sum of integer multiples of $1 \leq x_1, x_2, x_3 \leq X$.

Auto judge: [https://dmoj.ca/problem/neercnorthern07e](https://dmoj.ca/problem/neercnorthern07e) (NOTE: the constraints necessitate 64-bit integer types such as long long or Int64.)

5. Consider the standard value maximization knapsack setup: there are $n$ items, the $i^{th}$ of which has value $v_i$ and weight $w_i$. However, the $w_i$s may be negative, all we know is

$$-W \leq w_i \leq W$$

We want to pick a subset $S \subseteq [n]$ (without replacement) so that

$$\sum_{i \in S} w_i = 0$$

while maximizing the value of the items picked.

Give an algorithm that computes the maximum value with probability at least 0.9 in time $O(n^{1.5}W)$.  

2
Auto judge: https://dmoj.ca/problem/knapsack4

**HINT:** consider what happens to the optimum solution when the items are permuted randomly. You may find the following notes useful (specifically page 2) for proving the time complexity: http://math.mit.edu/~goemans/18310S15/chernoff-notes.pdf

The TAs are also aware of $O(nW)$ time deterministic algorithms for this type of problems (which are very very tricky).