Algorithms EECS 336 **Problem Set #2**
Date: Tuesday, January 15, 2019 **Due:** Tuesday, January 22, 2018

Submit your solution on Canvas.
Do not discuss these problems with other students. You should solve these problems on your own.

Problem 1. A trucking company wants to minimize the amount of money it spends on gas. In order to do so, it needs to develop a program that tells every driver where he or she should refuel the truck. In the beginning of every trip, the program receives a list of gas stations along the route. For each gas station, the list contains (a) the distance from the gas station to the beginning of the route; and (b) the price of gas at that gas station. The program also knows the length of the route, and the gas consumption per mile. The program needs to create a list of gas stations where the truck driver should stop for refueling, find the amount of fuel that driver should buy at each gas station, and calculate the total cost of the trip.

Assume that the capacity of the truck's gas tank is infinite. At each gas station, the driver can buy any amount of gas. For convenience, assume that the gas consumption of the vehicle is (exactly) 1 unit of gas per mile. At the beginning of the trip, the vehicle's gas tank is empty. It should also be empty at the end of the trip.

1. Describe a greedy algorithm that solves this problem.
2. Prove that your algorithm is correct.
3. Analyze the running time of your algorithm. To get a full credit for this problem, the running time your algorithm should be $O(n \log n)$.

Problem 2. In this exercise, you need to implement the greedy algorithm we discussed in class for interval scheduling on one machine. We ask you to write the following function.

- `int FindMaxSchedule (std::vector<std::pair<int,int>> intervals)`

Array `intervals` contains the set of intervals (the left endpoint of the i-th interval is `intervals[`i`].first`; the right endpoint of the i-th interval is `intervals[`i`].second`. This function should return the number of intervals in the optimal schedule.

Instructions for the programming assignment. Download files:

- `student_code_2.h` – this file should contain your solution.
- `problem_solver_2.cpp` – this is the main file in the project (don't edit this file!).
- `test_framework.h` – this is a library responsible for reading and writing data files (don't edit this file!)
- `problem_set_2.in` – this file contains test problems for your algorithm (don't edit this file!)

Place all files in a new folder/directory. Write your code in the function `FindMaxSchedule`. Also, write your name in the function `GetStudentName`. Both functions are located in file `student_code_2.h`. Compile and run your code. To compile your code do the following.

- If you use GNU C++ compiler, type
  `g++ -std=c++11 problem_solver_2.cpp -o problem_solver_2`
- If you use CLang compiler, type
  `clang++ -std=c++11 problem_solver_2.cpp -o problem_solver_2`
- If you use Microsoft Visual C++ compiler, start `Developer Command Prompt` and type
  `cl /EHsc problem_solver_2.cpp`

Your compiler should be compatible with `C++11`. If you work in TLab, you need to start developer tools first: Type

- `scl enable devtoolset-4 bash`

Once you compile your code, start your program. Type `./problem_solver_2` on Unix or Mac and `problem_solver_2.exe` on Windows. Make sure that the executable is located in the same folder as file `problem_set_2.in`. Your program will generate `solution_2.dat` that contains solutions to the problems from file `problem_set_2.in`. If your code works correctly, you will get the following message:

- Problem set 2. Your algorithm solved all test problems correctly. Congratulations!
- Don't forget to submit your source code and file `solution_2.dat` via Canvas.

If your code makes a mistake, you may get a message like this:

- `Problem set 2.  Mistake in problem #15.  Correct answer:  4.  Your answer:  12.`

Finally, when your code is ready, submit files `student_code_2.h` and `solution_2.dat` via Canvas. Make sure that you are submitting the latest versions.

Remark: If you want to debug your code, please, type `./problem_solver_2 15` on Unix or Mac and `problem_solver_2.exe 15` on Windows. This command will call your function only on one problem – the problem #15 and thus let you debug your code on the problem where your program erred. Note that this command will not generate or update `solution_2.dat`. So before submitting your solution, you need to run your program without any command line arguments.