CMPS 2433 Program 3
Greedy Scheduling
Due Nov 12, 2021 9 AM

Purpose: The purpose of this to implement a greedy algorithm to find an optimal (or near optimal) solution to a scheduling problem. To employ a sorting algorithm

Problem: Given a set of activities, along with a starting time and a time-length to complete the activity, find the maximum number of activities that can be performed by a single person, assuming that person can only work on a single activity at a time.

Method: Use a greedy algorithm to ***select the shortest activity that can be started after the previous activity has been completed***. Using a greedy approach will always result in an optimal solution to this problem. The idea is to initially compute the finish times, then *sort the activities* in increasing order of their finish times. Then create a set to store the selected activities and initialized it with the first activity. Then from the second activity onward, include the activity in the set if the activity's start time is greater than or equal to the finish time of the last selected activity. Do the sort of the data below on a piece of paper to check this out.

You may use the data structure of your choice – array, vector, linked list, set, etc. Use a `class` or use a `struct` to store an activity's data `(start_time, length, ending_time)`.

Input: The first line of input will contain the number of activities to process, N, where 1 <= N <=20.
The next N lines will contain each activities' information start_time and length.

Sample Input:

```
11
1    3
3    2
0    6
5    2
3    5
5    4
6    4
8    3
8    4
2    11
12   2
```

Corresponding Sample Output:

```
  Catherine Stringfellow
  Program 3: Greedy Scheduling/Sorting Program

  Start Time  Length    Finish Time
  1           3           4
  5           2           7
  8           3           11
  12          2           14

  Maximum number of activities scheduled is 4.
```

TURN IN:

- Upload .cpp file to D2L dropbox for Program 3
- Turn in on paper: .cpp file, your made up dataset, hand calculations showing two sorted lists of activities – one for my input and one for your input, and two outputs.