Rupinder Kaur

Nov 08, 2020

Part 5- Use an array based stack to solve the "longest increasing subsequence problem

In my program, first I added libraries and then I created a function to print the longest increasing subsequence (LIS) *void printLIS*. In which I used a foreach loop *for (int x : arr)*. Advantages of Foreach loop is 1) Makes code more readable. 2) Eliminates the possibility of programming errors. Then there is a print statement to print the result from the loop.

After that I used another function void constructPrintLIS. This function basically does the main part which is to construct and print LIS. In this function, first I created a 2d vector and then I used a push back statement where L[0] is equal to arr[0]. After that I used two for loops, in the first loop the index starts from 1 and in the second loop every j is less than i. Inside the second loop there is an if statement where j < i and arr[j] < arr[i] and if there is no such j then L[i] is equal to arr[i]. Following that there is another push back statement which shows L[i] ends with arr[i]. Then I initialize another 1d vector with data type and assigned variable. This vector helps in L[i] stores increasing subsequence of arr[0...i] that ends with arr[i]. After that I used a foreach loop where the longest increasing subsequence (LIS) will be max of all increasing subsequences of array. Then I add a print function where max will contain the longest increasing subsequence.

Now in the driver main function I declare an array value and add a construct function. After that the program will run properly. We define a vector L such that L[i] is itself a vector that stores the longest increasing subsequence of an array that ends with arr[i]. For example, for an array that I used [3, 2, 0, 4, 5, 1]. The way i understand it works something like this:

```
L[0] : 3

L[1]: 2

L[2]: 0

L[3]: 0 4

L[4]: 0 4 5  ----> OUTPUT

L[5]: 1
```