**Longest Increasing Subsequence.**

DESCRIPTION

**Objective:**

write a program to find the longest increasing subsequence.

**Background of the problem statement:**

Writing a Java code to find the longest increasing subsequence from a list of random numbers.

**You must use the following:**

* Eclipse/IntelliJ: An IDE to code for the application.
* Java: A programming language.
* Git: To connect and push files from the local system to GitHub.
* GitHub: To store the application code and track its versions.
* Core Java concepts: variables, data types, operators, type casting, control statements, class, objects, access specifiers, and core keywords like final, this, and static.

**Following requirements should be met:**

* The versions of the code should be tracked on GitHub repositories.
* The code should work properly for n numbers, where n<100.

**Steps needed to write a program to read , write and append to a file:**

1. Creating a new project in Eclipse

2. Writing a Java code to find the longest increasing subsequence from a list of random numbers.

3. Executing the program and verifying the result according to the particular operation.

4. Print the result.

5. Pushing the code to GitHub repositories.

**Step1:**

**Creating a new project in Eclipse:**

* Open Eclipse
* Go to File -> New -> Project -> Java Project -> Next.
* Type in project name as **Fourth-Project** and click on “Finish.”
* Select your project and go to File -> New -> Package.
* Enter **com.simplilearn.test** as Package name.
* Select your project and go to File -> New -> Class.
* Enter **LIS** as class name, check the checkbox “public static void main(String[] args)”., and click on “Finish.”

Now, let us discuss the Longest Increasing Subsequence (LIS) problem as an example problem that can be solved using Dynamic Programming.

The Longest Increasing Subsequence (LIS) problem is to find the length of the longest subsequence of a given sequence such that all elements of the subsequence are sorted in increasing order.

For example, the length of LIS for {10, 22, 9, 33, 21, 50, 41, 60} is 5 and LIS is {10, 22, 33, 50, 60}.

arr[ ] = {10, 22, 9, 33, 21, 50, 41, 60}

LIS = {10, 22, 33, 50, 60}

**Input:** arr[] = {3, 10, 2, 1, 20}

**Output:** Length of LIS = 3

The longest increasing subsequence is 3, 10, 20

**Input:** arr[] = {3, 2}

**Output:** Length of LIS = 1

The longest increasing subsequences are {3} and {2}

**Input:** arr[] = {50, 3, 10, 7, 40, 80}

**Output:** Length of LIS = 4

The longest increasing subsequence is {3, 7, 40, 80}

The simulation of approach will make things clear:

For Example:

Input : arr[] = {3, 10, 2, 11}

LIS[] = {1, 1, 1, 1} (initially)

**Iteration-wise simulation :**

1. arr[2] > arr[1] {LIS[2] = max(LIS [2], LIS[1]+1)= 2}
2. arr[3] < arr[1] {No change}
3. arr[3] < arr[2] {No change}
4. arr[4] > arr[1] {LIS[4] = max(LIS [4], LIS[1] +1)= 2}
5. arr[4] > arr[2] {LIS[4] = max(LIS [4], LIS[2] +1)= 3}
6. arr[4] > arr[3] {LIS[4] = max(LIS [4], LIS[3] +1)= 3}

lis[] returns the length of the longest increasing subsequence in arr[] of size n

**int** lis[]= **new** **int**[n];

**int** i, j, max = 0;

Initialize LIS values for all indexes

**for** (i = 0; i < n; i++)

lis[i] = 1;

Compute optimized LIS values in bottom-up manner

**for** (i = 1; i < n; i++)

**for** (j = 0; j < i; j++)

**if** (arr[i] > arr[j] && lis[i] < lis[j] + 1)

lis[i] = lis[j] + 1;

Pick maximum of all LIS values

**for** (i = 0; i < n; i++)

**if** (max < lis[i])

max = lis[i];

If the above loop terminates then

**return** max;

**Step 2:**

**Writing a Java code to find the longest increasing subsequence from a list of random numbers:**

**package** com.simplilearn.test;

**public** **class** LIS {

**static** **int** lis(**int** arr[], **int** n)

{

**int** lis[]= **new** **int**[n];

**int** i, j, max = 0;

**for** (i = 0; i < n; i++)

lis[i] = 1;

**for** (i = 1; i < n; i++)

**for** (j = 0; j < i; j++)

**if** (arr[i] > arr[j] && lis[i] < lis[j] + 1)

lis[i] = lis[j] + 1;

**for** (i = 0; i < n; i++)

**if** (max < lis[i])

max = lis[i];

**return** max;

}

**public** **static** **void** main(String args[])

{

**int** arr[] = { 10, 22, 9, 33, 21, 50, 41, 60 };

**int** n = arr.length;

System.***out***.println("Length of array is " + n );

System.***out***.println("Length of lis is " + *lis*(arr, n) + "\n");

}

}

**Step 3:**

**Executing the program and verifying the result according to the particular operation:**

Before executing the program, check for syntactical corrections. If no errors are found, follow the steps mentioned below:

* [*Right click*] in the program space
* Select *Run As Java Application*

**Step4:**

**Print the result:**

Length of array is 8

Length of lis is 5

The code is working properly for n numbers, where n<100.

**Step5:**

**Pushing the code to your GitHub repositories:**

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command: **git push -u origin master**