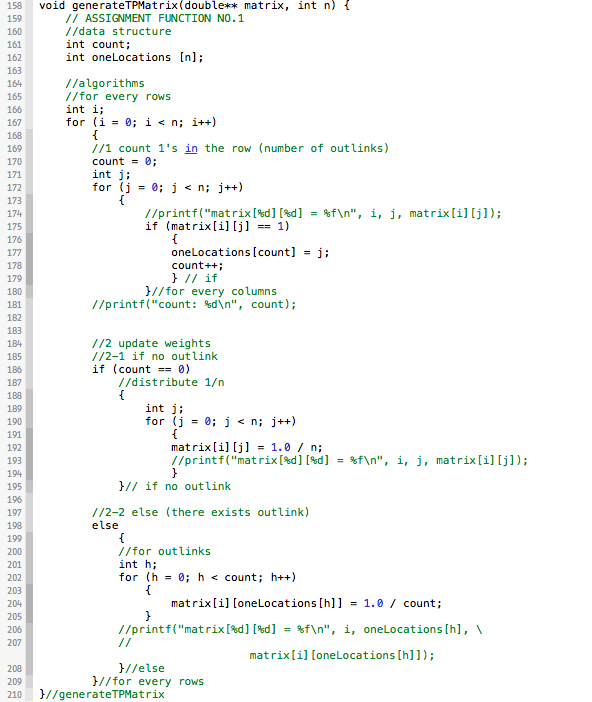
**Mini Project 2: Page Rank**

2011120355 Hee Kyung, Yoon

Table of Contents

1. Matrix Implementation
   1. generateTPMatrix: code, description
   2. calculatePageRank: code, description
   3. Screen Shots
2. List Implementation
   1. calculatePageRank: code, description
   2. Screen Shots
3. **Matrix Implementation**
   1. **generateTPMatrix**

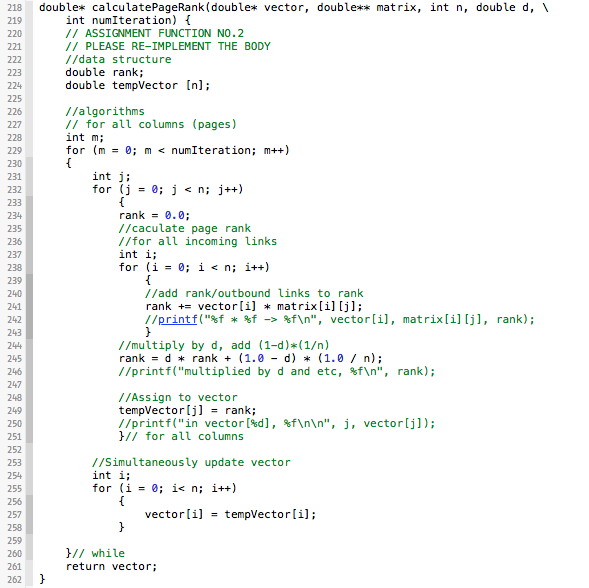
* **code**



* **description**

|  |
| --- |
| void generateTPMatrix (double\*\* matrix, int n)  1 for every rows in matrix  // count 1’s in the row (number of outlinks)  1 for every column  1 if 1 (outlink)  1 store the location  2 increment count  // update weights  1 if no outlink (count == 0)  1 for every column  1 assign 1.0 / n  2 else (count > 0)  1 for outlink columns (stored locations)  1 assign 1.0 / count |

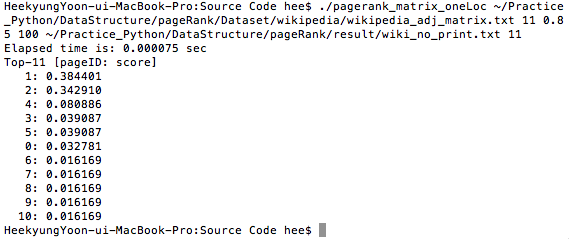
* 1. **calculatePageRank**
* **code**



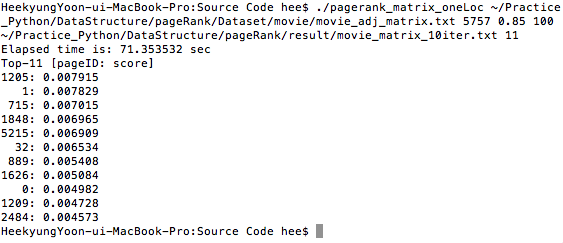
* **description**

|  |
| --- |
| double\* calculatePageRank(double\* vector, double\*\* matrix, int n, double d, \  int numIteration)  1 for numIterations  1 for every columns (pages)  // calculate page rank  1 for every inlink  1 rank += inlink’s rank / inlink’s outlinks  2 rank = d \* rank + (1.0 – d) \* (1.0 / n)  3 tempVector[i] = rank  // simultaneously update vertex  2 assign tempVector to vertex |

* 1. **Screen Shots**
* **wikipedia**

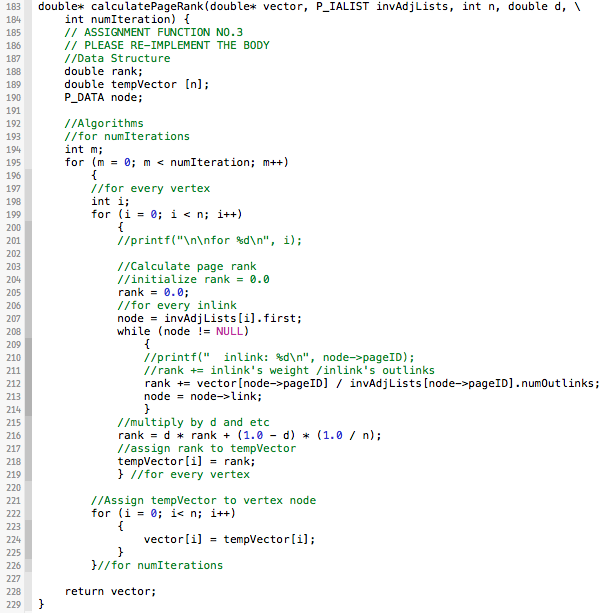


* **movie**



1. **List Implementation**
   1. **calculatePageRank**

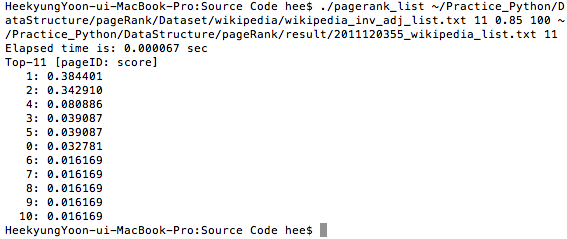
* **code**

****

* **description**

|  |
| --- |
| double\* calculatePageRank(double\* vector, P\_IALIST invAdjLists, int n, double d, \  int numIteration)  1 for numIterations  1 for every vertex  // calculate page rank  1 for every inlink  1 rank += inlink’s rank / inlink’s outlinks  2 rank = d \* rank + (1.0 – d) \* (1.0 / n)  3 tempVector[i] = rank  // simultaneously update vertex  2 assign tempVector to vertex |

* 1. **screenshots**
* **wikipedia**



* **movie**

