|  |  |
| --- | --- |
| Student Name | Saddar U Din |
| Roll Number | 21SW077 |
| Section # | 1 |
| Subject | DSA PBL |

**SolutionClass:**

*package* PBL;  
  
*import* java.io.File;  
*import* java.io.FileNotFoundException;  
*import* java.util.Scanner;  
  
  
*public class* Solutions {  
  
 *//University array has been created to store universities' objects* University[] universities=*new* University[100];*//values  
 //String array to store keys of every object of university* String[] keys=*new* String[100];  
  
 *//Constructor to fetch data from csv file to University array  
 //and txt file to keys array  
  
 //Hashmap: to store data in key value pair  
 private static final* MyHashTable ***map***=*new* MyHashTable(100);  
 *//stack1 for storing data order by noOfPublication  
 private static final* MyLinkedStack ***stack1***=*new* MyLinkedStack();  
 *//stack2 for storing data order by PakRanking  
 private static final* MyLinkedStack ***stack2***=*new* MyLinkedStack();  
 *private static final* MyLinkedStack ***SindhUETs***=*new* MyLinkedStack();  
 *private static final* MyLinkedStack ***PunjabUETs***=*new* MyLinkedStack();  
 *private static final* MyLinkedStack ***BalochistanUETs***=*new* MyLinkedStack();  
 *private static final* MyLinkedStack ***KPK\_UETs***=*new* MyLinkedStack();  
 *private static final* MyLinkedStack ***AJK\_UETs***=*new* MyLinkedStack();  
 *private static final* MyLinkedStack ***GilgitBiltistanUETs***=*new* MyLinkedStack();  
 *public void* readFile(){  
 *//Reading from csv file and txt file* File file=*new* File("PBL/Top 100 Unis of Pakistan.csv");  
 File file1=*new* File("PBL/Keys.txt");  
 *try* {  
 *int* i=0;  
 Scanner scan = *new* Scanner(file);  
 Scanner scanner=*new* Scanner(file1);  
 *while* (scan.hasNextLine()){  
 String[] parts= scan.nextLine().split(",");  
 universities[i]=*new* University(parts[0],Integer.*parseInt*(parts[1]),  
 Integer.*parseInt*(parts[2]),Integer.*parseInt*(parts[3]),  
 parts[4],parts[5],Integer.*parseInt*(parts[6]));  
 keys[i]=scanner.nextLine();  
 i++;  
 }  
 } *catch* (FileNotFoundException e) {  
 *throw new* RuntimeException(e);  
 }  
 }  
  
 *//to insert data into hashmap and also return map  
 public void* insertIntoMap(){  
 *for*(*int* i=0;i< universities.length;i++)***map***.put(keys[i],universities[i]);  
 }  
 *public void* insertIntoStack(String orderBy){  
 *for*(*int* i=0;i< keys.length;i++){  
 *if*(orderBy.equals("noOfPublications")) ***stack1***.push(keys[i],universities[i],orderBy);  
 *else if*(orderBy.equals("PakRanking")){  
 ***stack2***.push(keys[i],universities[i],orderBy);  
 *switch* (universities[i].getProvince()){  
 *case* "Sindh"->***SindhUETs***.push(keys[i],universities[i],orderBy);  
 *case* "Punjab"->***PunjabUETs***.push(keys[i],universities[i],orderBy);  
 *case* "Balochistan"->***BalochistanUETs***.push(keys[i],universities[i],orderBy);  
 *case* "KPK"->***KPK\_UETs***.push(keys[i],universities[i],orderBy);  
 *case* "AJK"->***AJK\_UETs***.push(keys[i],universities[i],orderBy);  
 *case* "Gilgit Biltistan"->***GilgitBiltistanUETs***.push(keys[i],universities[i],orderBy);  
 }  
 }  
 }  
  
 }  
 *public static* MyHashTable getMap(){*return* ***map***;}  
  
 *public static* MyLinkedStack getStack1(){*return* ***stack1***;}  
 *public static* MyLinkedStack getStack2(){*return* ***stack2***;}  
  
 *public static* MyLinkedStack getSindhUETs() {  
 *return* ***SindhUETs***;  
 }  
  
 *public static* MyLinkedStack getPunjabUETs() {  
 *return* ***PunjabUETs***;  
 }  
  
 *public static* MyLinkedStack getGilgitBiltistanUETs() {  
 *return* ***GilgitBiltistanUETs***;  
 }  
  
 *public static* MyLinkedStack getKPK\_UETs() {  
 *return* ***KPK\_UETs***;  
 }  
  
 *public static* MyLinkedStack getBalochistanUETs() {  
 *return* ***BalochistanUETs***;  
 }  
  
 *public static* MyLinkedStack getAJK\_UETs() {  
 *return* ***AJK\_UETs***;  
 }  
  
 *//Problem1  
 public* University search(MyHashTable map,String key){  
 *return* map.get(key);  
 }*//searching from hashtable  
 public* University[] topX(MyLinkedStack stack,*int* X){  
 University[] uet=*new* University[X];  
 String[] keys=*new* String[X];  
 MyHashTable map=*getMap*();  
 *for*(*int* i=0;i<X;i++){  
 keys[i]=stack.pop();  
 uet[i]=map.get(keys[i]);  
 }  
 *for*(*int* i=X-1;i>=0;i--)stack.push(keys[i]);  
 *return* uet;  
 }  
 *public* University[] bottomX(MyLinkedStack stack,*int* X){  
 University[] uet=*new* University[X];  
 MyHashTable map=*getMap*();  
 String[] keys=*new* String[map.size()];  
 *int* count=0;  
 *for*(*int* i=0;i< map.size();i++) keys[i]=stack.pop();  
 *for*(*int* i=map.size()-X;i< map.size();i++)uet[count++]=map.get(keys[i]);  
 *for*(*int* i=99;i>=0;i--)stack.push(keys[i]);  
 *return* uet;  
 }  
 *public* University percentile(*int* percentile,MyLinkedStack stack){  
 */\*  
 The percentile formula determines the performance of anything  
 over other things of same class.  
 For example: The percentile formula is used in finding where a  
 student stands in the test compared to other candidates.  
 A percentile is a number where a certain percentage of scores  
 fall below the given number.  
 \*/  
 /\*  
 formula : n=P/100\*N where  
 n=ordinal rank  
 N=No of values in data;  
 P=percentile  
 \*/  
 int* position=(percentile\*stack.size())/100;  
 *return getMap*().get(stack.get(position));  
 }  
 *public* String performance(){  
 String x="",y= "";  
 *int* punjabPub=0,sindhPub=0,balPub=0,kpkPub=0,ajkPub=0,gilgitPub=0;  
 MyLinkedStack stack1=*getSindhUETs*();  
 MyLinkedStack stack2=*getPunjabUETs*();  
 MyLinkedStack stack3=*getBalochistanUETs*();  
 MyLinkedStack stack4=*getKPK\_UETs*();  
 MyLinkedStack stack5=*getAJK\_UETs*();  
 MyLinkedStack stack6=*getGilgitBiltistanUETs*();  
 *for*(*int* i=1;i<=stack1.size();i++)  
 sindhPub+=*getMap*().get(stack1.get(i)).getNoOfPublications();  
 *for*(*int* i=1;i<=stack2.size();i++)  
 punjabPub+=*getMap*().get(stack2.get(i)).getNoOfPublications();  
 *for*(*int* i=1;i<=stack3.size();i++)  
 balPub+=*getMap*().get(stack3.get(i)).getNoOfPublications();  
 *for*(*int* i=1;i<=stack4.size();i++)  
 kpkPub+=*getMap*().get(stack4.get(i)).getNoOfPublications();  
 *for*(*int* i=1;i<=stack5.size();i++)  
 ajkPub+=*getMap*().get(stack5.get(i)).getNoOfPublications();  
 *for*(*int* i=1;i<=stack6.size();i++)  
 gilgitPub+=*getMap*().get(stack6.get(i)).getNoOfPublications();  
 *double* avg1=(*double*) sindhPub/stack1.size();  
 *double* avg2=(*double*) punjabPub/stack1.size();  
 *double* avg3=(*double*) balPub/stack1.size();  
 *double* avg4=(*double*) kpkPub/stack1.size();  
 *double* avg5=(*double*) ajkPub/stack1.size();  
 *double* avg6=(*double*) gilgitPub/stack1.size();  
 *double* best,worst;  
 best= Math.*max*(avg1, avg2);  
 best= Math.*max*(best, avg3);  
 best= Math.*max*(best, avg4);  
 best= Math.*max*(best, avg5);  
 best= Math.*max*(best, avg6);  
 worst= Math.*min*(avg1, avg2);  
 worst= Math.*min*(worst, avg3);  
 worst= Math.*min*(worst, avg4);  
 worst= Math.*min*(worst, avg5);  
 worst= Math.*min*(worst, avg6);  
 *if* (best == avg1) x = "Sindh";  
 *else if* (best == avg2) x = "Punjab";  
 *else if* (best == avg3) x = "Balochistan";  
 *else if* (best == avg4) x = "KPK";  
 *else if* (best == avg5) x = "AJK";  
 *else if* (best == avg6) x = "Gilgit";  
 *if* (worst == avg1) y = "Sindh";  
 *else if* (worst == avg2) y = "Punjab";  
 *else if* (worst == avg3) y = "Balochistan";  
 *else if* (worst == avg4) y = "KPK";  
 *else if* (worst == avg5) y = "AJK";  
 *else if* (worst == avg6) y = "Gilgit";  
  
  
  
 *return* "Best Province: "+x+"\n Worst Province: "+y;  
 }  
  
 *public static void* main(String[] args) {  
 Solutions s=*new* Solutions();  
 s.readFile();  
 s.insertIntoMap();  
 s.insertIntoStack("noOfPublications");  
 s.insertIntoStack("PakRanking");  
 *while* (*true*) {  
 System.***out***.println("\n\n\n");  
 System.***out***.println("1.Search any university by providing key");  
 System.***out***.println("2. Top x universities of Pakistan");  
 System.***out***.println("3. n universities from bottom");  
 System.***out***.println("4. nth Percentile university");  
 System.***out***.println("5. See provincial ranking of the universities");  
 System.***out***.println("6. Best and worst province");  
 System.***out***.println("7.Exit");  
 Scanner scan = *new* Scanner(System.***in***);  
 *switch* (scan.next()) {  
 *case* "1" -> {  
 System.***out***.println("Enter four word key: ");  
 String str=scan.next();  
 System.***out***.println("wRank\t aRank\t pRank\tPub\t\t\t\t\t\t\tLoc\t\t\t\t\t\t\tProvince\t\t\t\t\t\t\t\t\t\tName");  
 System.***out***.println(*getMap*().get(str));  
 }  
 *case* "2" -> {  
 System.***out***.println("Enter number of universities: ");  
 *int* b = Integer.*parseInt*(scan.next());  
 System.***out***.println("On the basis of: ");  
 System.***out***.println("1> Number of publication ");  
 System.***out***.println("2> Pakistani Ranking ");  
 System.***out***.println("Your choice: ");  
 String a = scan.next();  
 System.***out***.println("wRank\t aRank\t pRank\tPub\t\t\t\t\t\t\tLoc\t\t\t\t\t\t\tProvince\t\t\t\t\t\t\t\t\t\tName");  
 *if* (a.equals("1"))  
 *for* (University u : s.topX(*getStack1*(), b))  
 System.***out***.println(u);  
 *else if* (a.equals("2"))  
 *for* (University u : s.topX(*getStack2*(), b))  
 System.***out***.println(u);  
 }  
 *case* "3" -> {  
 System.***out***.println("Enter number of universities: ");  
 *int* b = Integer.*parseInt*(scan.next());  
 System.***out***.println("On the basis of: ");  
 System.***out***.println("1> Number of publication ");  
 System.***out***.println("2> Pakistani Ranking ");  
 System.***out***.println("Your choice: ");  
 String a = scan.next();  
 System.***out***.println("wRank\t aRank\t pRank\tPub\t\t\t\t\t\t\tLoc\t\t\t\t\t\t\tProvince\t\t\t\t\t\t\t\t\t\tName");  
 *if* (a.equals("1"))  
 *for* (University u : s.bottomX(*getStack1*(), b))  
 System.***out***.println(u);  
 *else if* (a.equals("2"))  
 *for* (University u : s.bottomX(*getStack2*(), b))  
 System.***out***.println(u);  
 }  
 *case* "4" -> {  
 System.***out***.println("Enter percentile: ");  
 *int* b = Integer.*parseInt*(scan.next());  
 System.***out***.println("On the basis of: ");  
 System.***out***.println("1> Number of publication ");  
 System.***out***.println("2> Pakistani Ranking ");  
 System.***out***.println("Your choice: ");  
 String a = scan.next();  
 System.***out***.println("wRank\t aRank\t pRank\tPub\t\t\t\t\t\t\tLoc\t\t\t\t\t\t\tProvince\t\t\t\t\t\t\t\t\t\t\tName");  
 *if* (a.equals("1"))  
 System.***out***.println(s.percentile(b, *getStack1*()));  
 *else if* (a.equals("2"))  
 System.***out***.println(s.percentile(b, *getStack2*()));  
 }  
 *case* "5" -> {  
 System.***out***.println("1.Top Ranked Universities of Sindh");  
 System.***out***.println("2.Top Ranked Universities of Punjab");  
 System.***out***.println("3.Top Ranked Universities of Balochistan");  
 System.***out***.println("4.Top Ranked Universities of KPK");  
 System.***out***.println("5.Top Ranked Universities of AJK");  
 System.***out***.println("6.Top Ranked Universities of Gilgit Biltistan");  
 String c = scan.next();  
 System.***out***.println("wRank\t aRank\t pRank\tPub\t\t\t\t\t\t\tLoc\t\t\t\t\t\t\tProvince\t\t\t\t\t\t\t\t\t\tName");  
 *switch* (c) {  
 *case* "1" -> *getSindhUETs*().traverse();  
 *case* "2" -> *getPunjabUETs*().traverse();  
 *case* "3" -> *getBalochistanUETs*().traverse();  
 *case* "4" -> *getKPK\_UETs*().traverse();  
 *case* "5" -> *getAJK\_UETs*().traverse();  
 *case* "6" -> *getGilgitBiltistanUETs*().traverse();  
 }  
 }  
 *case* "6" -> System.***out***.println(s.performance());  
 *case* "7"->System.*exit*(0);  
 }  
 }  
 }  
  
}

**University Class:**

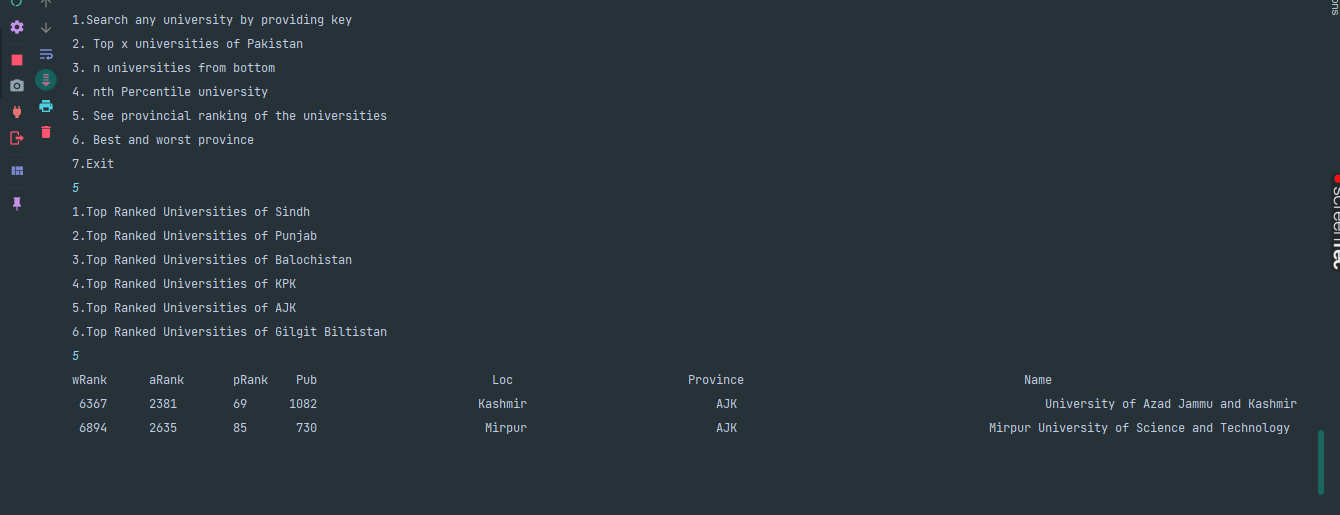
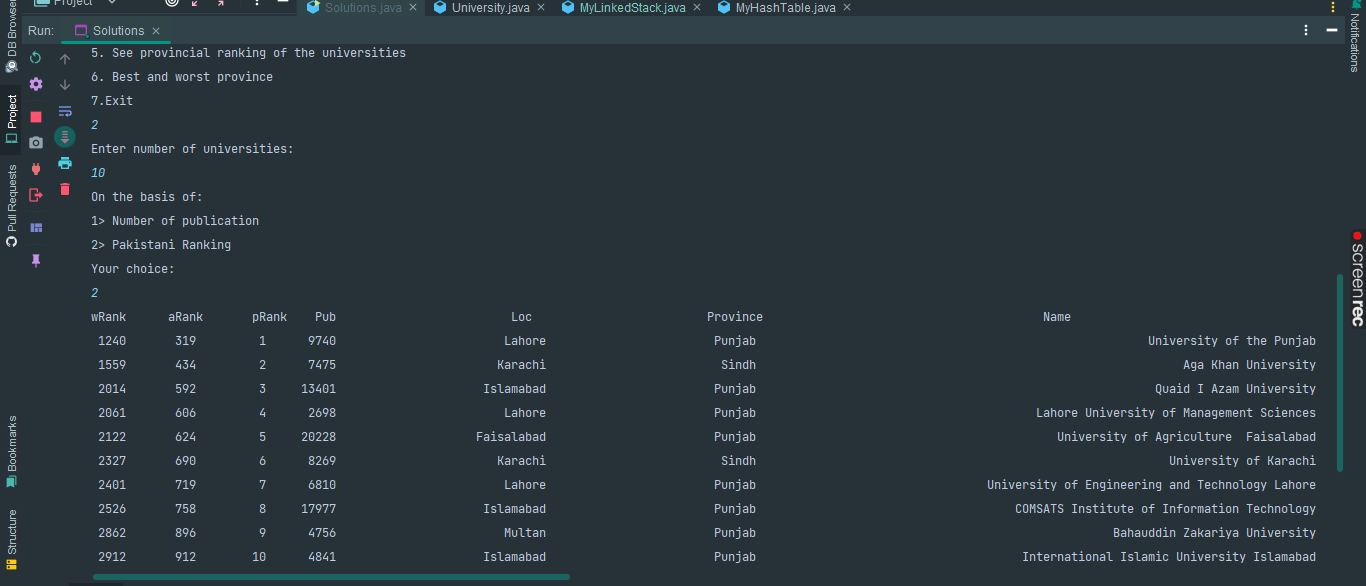
*package* PBL;  
  
*public class* University {  
 *private* String name,location,province;  
 *private int* worldRanking,asiaRanking,pakRanking,noOfPublications;  
  
 *public* University(String name,*int* worldRanking,*int* asiaRanking,*int* pakRanking,String location,String province,*int* noOfPublications){  
 *this*.name=name;  
 *this*.worldRanking=worldRanking;  
 *this*.asiaRanking=asiaRanking;  
 *this*.pakRanking=pakRanking;  
 *this*.location=location;  
 *this*.province=province;  
 *this*.noOfPublications=noOfPublications;  
 }  
  
 *public* University(){  
  
 }  
  
 *public* String getName() {  
 *return* name;  
 }  
  
 *public* String getLocation() {  
 *return* location;  
 }  
  
 *public* String getProvince() {  
 *return* province;  
 }  
  
 *public int* getWorldRanking() {  
 *return* worldRanking;  
 }  
  
 *public int* getAsiaRanking() {  
 *return* asiaRanking;  
 }  
  
 *public int* getPakRanking() {  
 *return* pakRanking;  
 }  
  
 *public int* getNoOfPublications() {  
 *return* noOfPublications;  
 }  
  
 @Override  
 *public* String toString() {  
 *return* String.*format*("%5s%10s%10s%10s%30s%30s%80s",worldRanking,asiaRanking,pakRanking,noOfPublications,location,province,name);  
 }  
}

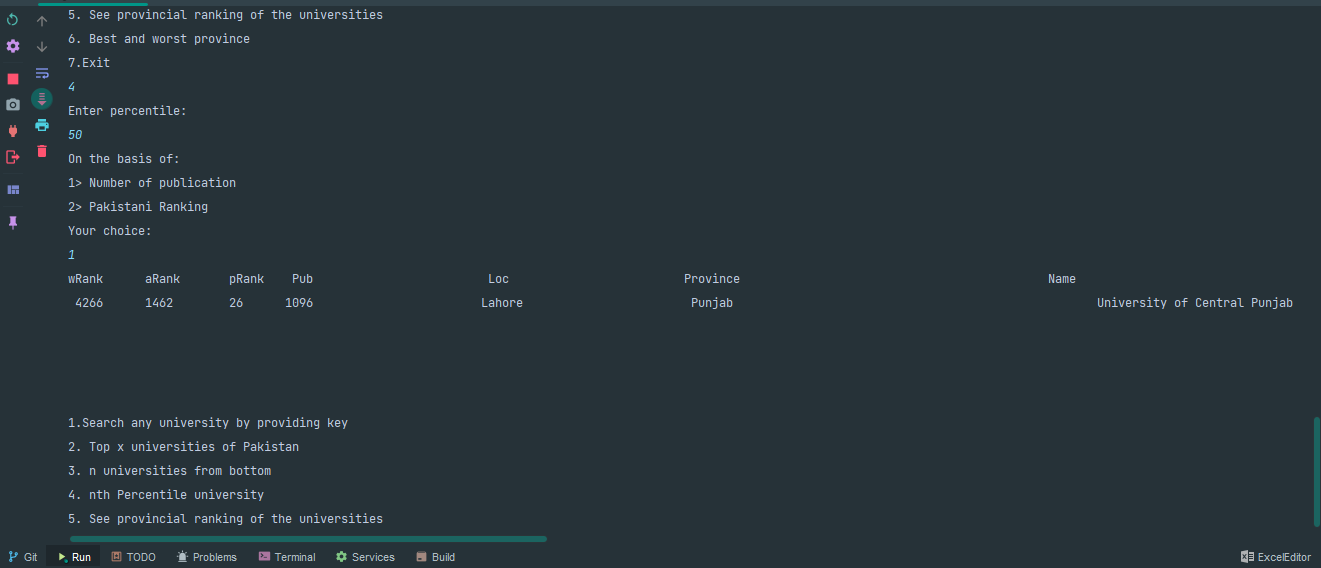
**Linked Stack Class:**

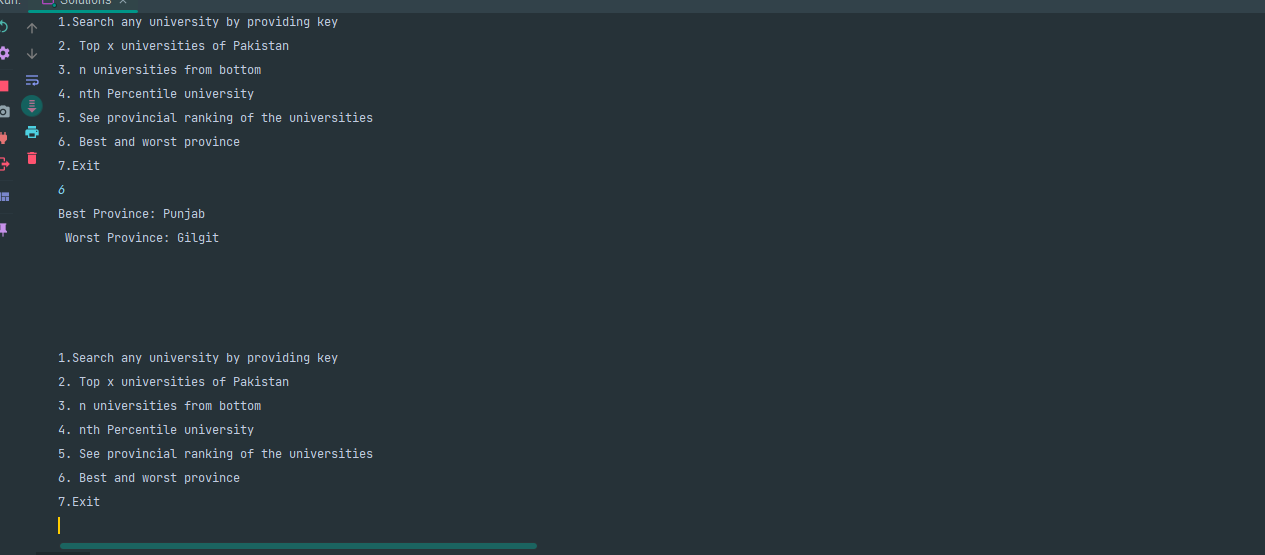
*package* PBL;  
  
*import* java.util.EmptyStackException;  
  
*class* MyLinkedStack {  
 *private static class* Node{  
 String data;  
 Node next;  
 *public* Node(String data,Node n){  
 *this*.data=data;  
 *this*.next=n;  
 }  
 }  
 *private* Node top;  
 *private int* size;  
 *public boolean* isEmpty(){  
 *return* size==0;  
 }  
 *public int* size() {  
 *return* size;  
 }  
  
 *public void* push(String data) {  
 top=*new* Node(data,top);  
 size++;  
 }  
 *public* String pop() {  
 *if*(isEmpty())*throw new* EmptyStackException();  
 String temp=top.data;  
 top=top.next;  
 size--;  
 *return* temp;  
 }  
 *public* String peak() {  
 *if*(isEmpty())*throw new* EmptyStackException();  
 *return* top.data;  
 }  
 *public void* push(String key,University university,String orderBy){  
 Node temp=top;  
 *if*(orderBy.equals("noOfPublications")){  
 *if*(isEmpty()|| Solutions.*getMap*().get(top.data).getNoOfPublications()< university.getNoOfPublications()){  
 top=*new* Node(key,top);  
 size++;  
 *return*;  
 }  
  
 *while* (temp.next!=*null*){  
 *if*(Solutions.*getMap*().get(temp.next.data).getNoOfPublications()< university.getNoOfPublications())*break*;  
 temp=temp.next;  
 }  
  
 }  
 *else if*(orderBy.equals("PakRanking")){  
 *if*(isEmpty()||Solutions.*getMap*().get(top.data).getPakRanking()> university.getPakRanking()){  
 top=*new* Node(key,top);  
 size++;  
 *return*;  
 }  
  
 *while* (temp.next!=*null*){  
 *if*(Solutions.*getMap*().get(temp.next.data).getPakRanking()> university.getPakRanking())*break*;  
 temp=temp.next;  
 }  
  
  
 }  
 Node n=*new* Node(key,*null*);  
 n.next=temp.next;  
 temp.next=n;  
 size++;  
  
 }  
 *public* String get(*int* position){  
 *if*(position<1||position>size)*throw new* IllegalArgumentException("Incorrect index");  
 Node temp=top;  
 *for*(*int* j=1;j<position;j++)temp=temp.next;  
 *return* temp.data;  
 }  
 *public void* traverse(){  
 *for*(Node n=top;n!=*null*;n=n.next) System.***out***.println(Solutions.*getMap*().get(n.data));  
 }  
  
}

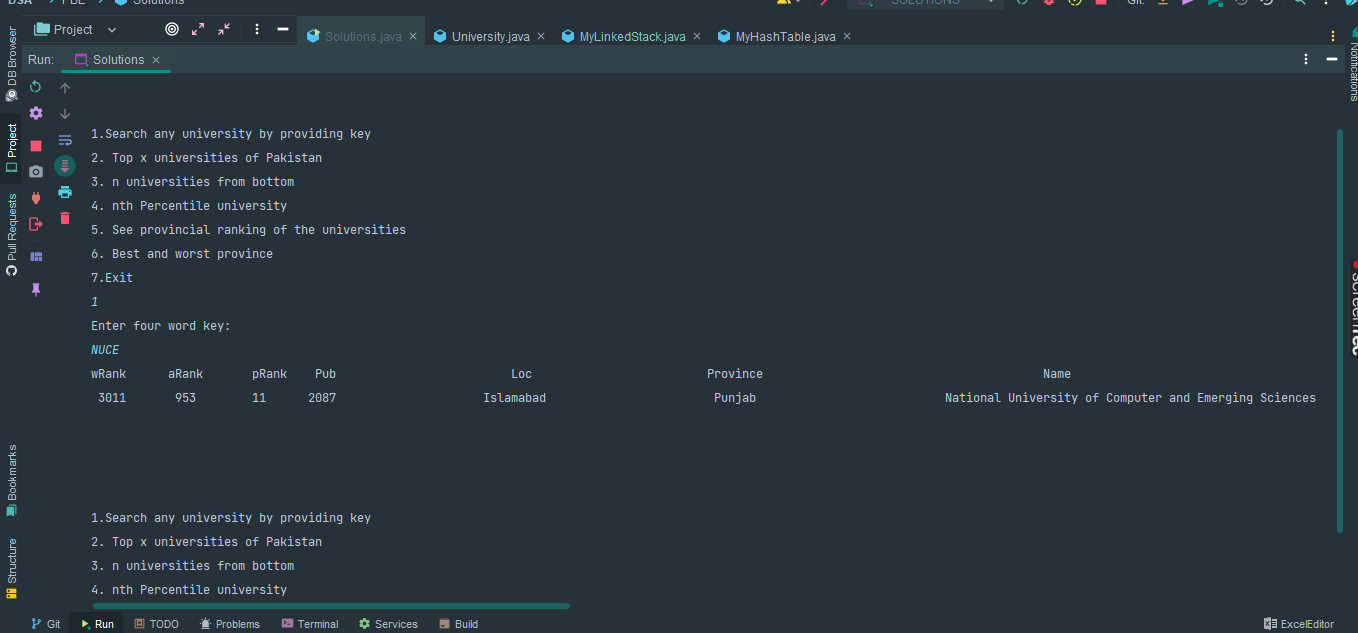
**HashTable class:**

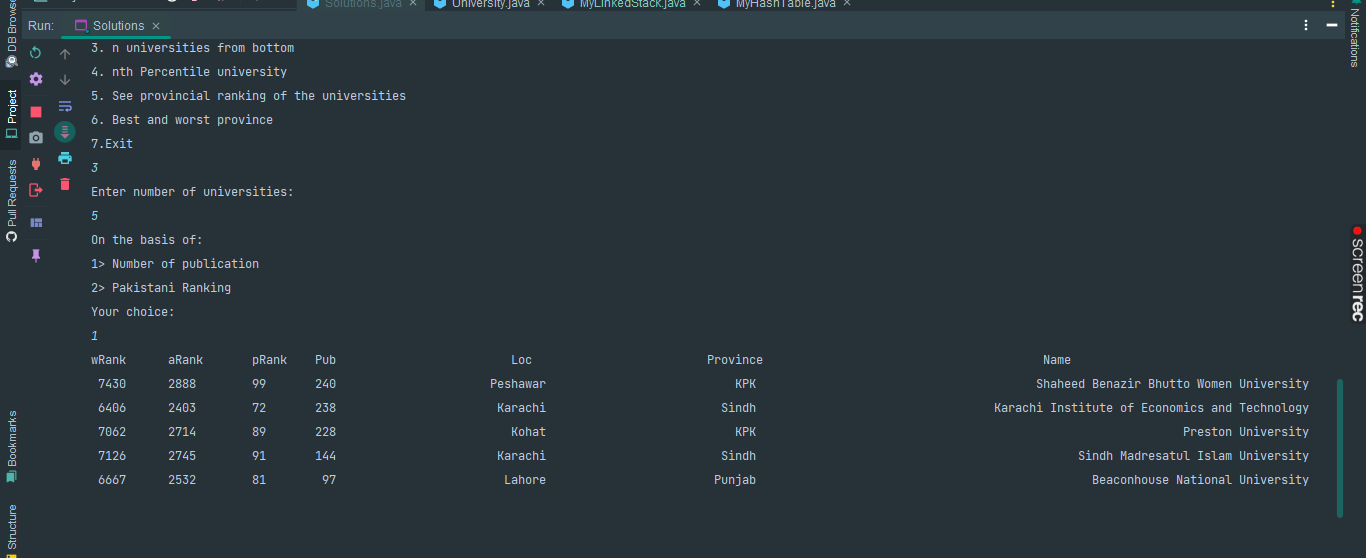
*package* PBL;  
  
  
*public class* MyHashTable {  
 *private static class* Node{  
 String key;  
 University value;  
 Node next;  
 }*//Node class to store key value pair* Node[] table;  
 *private int* size;  
 *public* MyHashTable(){  
 table=*new* Node[64];  
 }*//default constructor to make table of capacity of 64  
 public* MyHashTable(*int* capacity){  
 table=*new* Node[capacity];  
 }*//to make table of capacity provided by user(>0)  
 private int* hash(Object key){  
 *return* Math.*abs*(key.hashCode())% table.length;  
 }*//to make hash code of given key  
 private void* resize(){  
 Node[] newTable=table;  
 table=*new* Node[table.length\*2];  
 *for*(Node head:newTable){  
 *while* (head!=*null*){  
 Node temp=head.next;  
 *int* index=hash(head.key);  
 head.next=table[index];  
 table[index]=head;  
 head=temp;  
 }  
  
 }  
 }*//resize the table when size in more than 75%  
 public void* put(String key,University value){  
 *int* index=hash(key);  
 Node n=table[index];  
 *while* (n!=*null*){  
 *if*(n.key.equals(key))*break*;  
 n=n.next;  
 }  
 *if*(n!=*null*){  
 n.value=value;  
 *return*;  
 }  
 *if*(size>=0.75\*table.length)resize();  
 Node head=*new* Node();  
 head.value=value;  
 head.key=key;  
 head.next=table[index];  
 table[index]=head;  
 size++;  
 }  
 *public* University get(String key){  
 *int* index=hash(key);  
 Node head=table[index];  
 *while* (head!=*null*){  
 *if*(head.key.equals(key))*return* head.value;  
 head=head.next;  
 }  
 *return null*;  
 }  
 *public* University remove(String key){  
 *int* index=hash(key);  
 Node head=table[index];  
 *if*(head==*null*)*return null*;  
 *if*(head.key.equals(key)){  
 table[index]=head.next;  
 size--;  
 *return* head.value;  
 }  
 Node pre=head;  
 Node curr=pre.next;  
 *while* (curr!=*null*){  
 *if*(curr.key.equals(key)){  
 pre.next=curr.next;  
 size--;  
 *return* curr.value;  
 }  
 pre=curr;  
 curr=pre.next;  
 }  
 *return null*;  
 }  
 *public int* size(){*return* size;}  
 *public boolean* containsKey(String key){  
 *int* index=hash(key);  
 Node head=table[index];  
 *while* (head!=*null*){  
 *if* (head.key.equals(key))*return true*;  
 head=head.next;  
 }  
 *return false*;  
 }  
}

**Output Snippets: **

****

****

****

****

ALGORITHM

Step1:  
->Make University class with appropriate attributes.  
->Make two constructors an empty constructor and an argumentative constructor.  
->Make getters for all the attributes.  
->Make toString method.  
->Make a txt file and write (4 word) keys in it.  
->Make String array to store keys in Main class and University array to store Objects of University class.  
->Now make a method to read both the files .csv file and .txt file and store them in the arrays declared above.  
  
  
Step 2:  
->Make hashTable class with its appropriate methods  
->Now using any loop say for loop to take key and corresponding value(University) from their arrays  
 declared in the previous step and put them in hashTable one by one.  
  
  
Step 3:  
->Make Stack class with its appropriate methods also a method called get() that gets position of key  
 and returns key of that position if it exists, make another push() method that takes three arguments  
 1.orderBy(pakRanking/noOfPublications),2.Key to store in that stack and 3.University object to get noOfPublications  
 so that can be compared with other universities whose keys are already stored in the stack.  
  
->Now in the argumentative push method before store key decide its position by comparing with all  
 the universities and store that key.  
  
-> Now in main class make a method and using for loop call argumentative push method and pass the appropriate arguments.  
  
  
Step 4:  
->Use step three  
->Only difference is you have to pass the argument on which ranking you want to store like pakRanking etc.  
  
  
Problem 1:  
->To access the data without collision better to make the hashTable which stores LinkedList at every index  
->Now you can get the object by using appropriate index (hashcode) and traverse the linked list at that index.  
  
  
Problem 2:  
->Make a method that takes argument orderBy and pops the top x universities from the LinkedLists  
 based on the argument passed, stores them into array and push again into stack and return that array  
  
->Make method like previous but this time it stores x universities from bottom (use for loop to reach the position  
 from where you want to pop up the keys) stores them into array and then push again into the same stack at the end  
 and return that array  
  
->And nth percentile can be calculated by using formula : n=P/100\*N where: n=ordinal rank, P=percentile, N=No of values in data;  
 Through this formula calculate the percentile that will be the position of the university and then take key of that position from  
 stack and give it to the hashTable it will return you the information about the university i.e. University object.  
  
  
  
Problem 3:  
->Make stacks one for each province and store universities in respective stacks.  
->Now traverse each stack and calculate the average of no of publications of each stack.  
->Now the average will decide which province is best and which is the worst.