Sun Yat-Sen University
Data Structures and Algorithms
Spring 2016 Final Practical Examination

Date: Friday, June 17, 2016 Time: 13:30–15:30

About submission: Fill the source codes (with some comments) of the following 4 problems into the answer sheet. For each problem you should also submit an executable program. You can put the final answer sheet and the programs into one compressed package, and submit the package on the E-learning platform.

Problem 1 (20 points):

You need to order a list of pairs, v2, in the lexicographical order, that is, the list of pairs are ordered by the first component, and if the first components are the same, then they are ordered on the second component. Your program should use the following example:

```
 int a[] = {1, 3, 5, 6, 7, 4, 3, 4, 3, 2, 1,3,5,2,2,1, 2,1,2,3,2};
 const int N = sizeof(a) / sizeof(int);
 vector<pair<int, int>> v2;
 for (int i=0; i<N; i++)
     v2.push_back(make_pair(a[i], i));
  // v2 = (1,0), (3,1), (5,2), (6,3), (7,4),(4,5), (3,6), (4,7),(3,8),(2,9) , ...
  //How do you sort v2 on the first and second components? You need to define some functional object.
```

Problem 2 (25 points):

Given an adjacency list representation of a graph, output all the degrees of the vertices. Your program should have a main function in which an example is tested. You can use the following code and example:

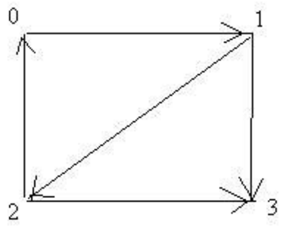
```
struct ALGraph{
    vector<list<int>> adj; //adjacency list of the graph
    int vexnum; // number of vertices
    int arcnum; //number of edges
    //A simple initialization.
  ALGraph(int n=0):vexnum(n){
        list<int> l;
        adj.resize(n,l);
  }
} ;

vector<pair<int, int> > degree(const ALGraph &g);
// returns a list of indegrees and outdegrees for all vertices

ALGraph mkALGraph(const char * f);
/* It returns an object of ALGraph given by the file f which contains the adjacency list of a graph.
   The first line is the number of vertices.
   The next n lines is the adjacency list. The first number is the index of the vertex, then its adjacency
vertices followed and -1 signals the end of the list of adjacency list. For example, the following gives a
```

graph and its representation:

```
4
0  1  -1
1  2  3  -1
2  0  3  -1
3  -1
```



The function returns an object of type ALGraph representing the graph.*/

Problem 3 (30 points):

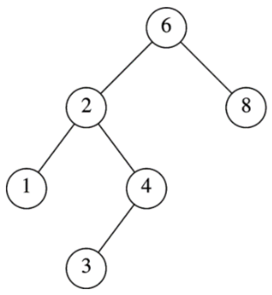
Define the following type for binary trees.

```
template <typename T> struct BinaryNode{
   T elem;
   BinaryNode *left;
   BinaryNode * right;
   BinaryNode(T d, BinaryNode *l=NULL, BinaryNode *r=NULL): elem(d),left(l),right(r){ };
};
```

Your task is to implement the inorder traversal using both recursion and non-recursion.

```
template <typename T>
void inorder_recursive(BinaryNode<T>* root, void (*visit)(T &x))
template <typename T>
void inorder(BinaryNode<T>* root, void (*visit)(T &x))
```

Your program should have a main function and test the example tree below:



Problem 4 (25 points):

Problem 2.22 in page 32 of the practical textbook. Note that you should submit the program with the name of “count.exe”, which should be run in the command line with the following way:

```
>count inputfile outputfile
```

You can use the following file to test your program:

Filename: Armstrong.txt

File contents:

```
I see trees of green, red roses too
I see them bloom for me and you
And I think to myself what a wonderful world.
```