

```
using System;
using System.IO;
using System.Collections;
using System.Text.RegularExpressions;

namespace english_find_words
{
    /// <summary>
    /// Summary description for Class1.
    /// </summary>
    ///
    class holder : IComparable
    {
        public int count;
        public string text;

        public holder(int c, string s)
        {
            this.count = c;
            this.text = s;
        }

        public int CompareTo(Object h)
        {
            int x = this.count.CompareTo(((holder) h).count);
            if(x > 0)
            {
                return -1;
            }
            else if( x == 0)
            {
                return 0;
            }
            else
            {
                return 1;
            }
            //return this.text.CompareTo(((holder) h).text);
        }
    }
    class Class1
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main(string[] args)
        {
            StreamReader f = File.OpenText(args[0]);
            string text = f.ReadToEnd();

            // Tokenize!
            string [] tokens = Regex.Split(text, "[^a-zA-Z']");
            int sub_len = Int32.Parse(args[1]);
            Hashtable map = new Hashtable();

            for(int i = 0; i < tokens.Length; i++)
            {
                foreach(string s in getSome(sub_len, i, tokens))
                {
                    if(!map.Contains(s.Trim()))
                    {
                        map[s.Trim()] = 1;
                    }
                    else
                }
            }
        }
    }
}
```

```
        {
            map[s.Trim()] = (int) map[s.Trim()] + 1;
        }
    }

    // Sort
    ArrayList l = new ArrayList();
    IDictionaryEnumerator ii = map.GetEnumerator();
    while(ii.MoveNext())
    {
        l.Add(new holder((int) ii.Value, (string) ii.Key));
    }
    l.Sort();

    ArrayList ll = new ArrayList();
    char [] sep = {' '};
    for(int i = 1; i <= sub_len; i++)
    {
        int added = 0;
        ArrayList temp = new ArrayList();
        for(int j = 0; j < l.Count && added < 1000; j++)
        {
            holder h = (holder) l[j];
            if(h.text.Split(sep).Length == i)
            {
                temp.Add(h);
                added++;
            }
        }
        ll.Add(temp);
    }
    // Output
    for(int i = 0; i < Int32.Parse(args[2]); i++)
    {
        foreach(ArrayList temp in ll)
        {
            holder h = (holder) temp[i];
            Console.Write(h.text+"\t"+h.count+"\t");
        }
        Console.WriteLine("");
    }
}

static string [] getSome(int num, int pos, string [] toks)
{
    num = (toks.Length - pos) < num ? toks.Length - pos : num;
    string [] temp = new string [num];
    string bit = "";
    for(int i = 0; i < num; i++)
    {
        bit += (" "+toks[pos + i].ToLower());
        bit = bit.Trim();
        temp[i] = bit;
    }

    return temp;
}
}
```