Name: Yu Qin

Results Screenshots

==========start of the write-up==========

Lab 1: K-Cluster

Code:

```
138 void HierarchicalClustering(){
139     InitialDistanceTable();
140     while (dataset.size() > n_clusters){
141         countcluster++;
142         cout<<"\ncount cluster times: "<<countcluster<<endl;
143         //find and merge two closest clusters, by shortest distance in dTable;
144         float minDt = INT_MAX;
145         int mi, mj;
146         FindClosestCluster(mi, mj, minDt);
147         ///cout<<mi<<"-mi,   mj:"<<mj<<endl;
148         //merge mj into mi;
149         MergeMJtoMI(mi, mj);
150         ///cout<<mi<<"-mi,   mj:"<<mj<<endl;
151         //rm the originial mj in dataset;
152         ///cout<<mi<<"-mi,   mj:"<<mj<<endl;
153         RemoveMJinDataset(mj);
154         //update dTable;
155         UpdateDistanceTable(mi,mj);
156         //rm row of mj and column of mj in dTable;
157         RemoveMJindTable(mj);
158     }
159 }
160 int main() {
161     //string infile = "input.txt";
162     //string infile = "spiral_format.txt";
163     //string infile = "aggregation_format.txt";
164     string infile = "Pathbased_format.txt";
165     dataset = MyReadFile(infile);
166     HierarchicalClustering();
167     //save result as txt in the format of ;
168     SaveResult();
169     return 0;
170 }
171
```

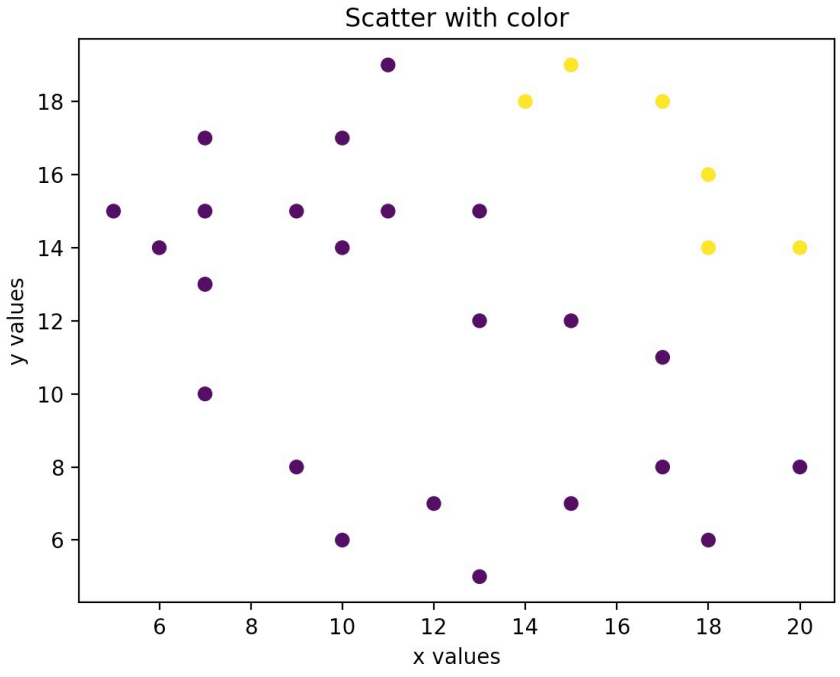
test case 1 and 2: with the std input given by prof, but different clusters of 2 and 3:

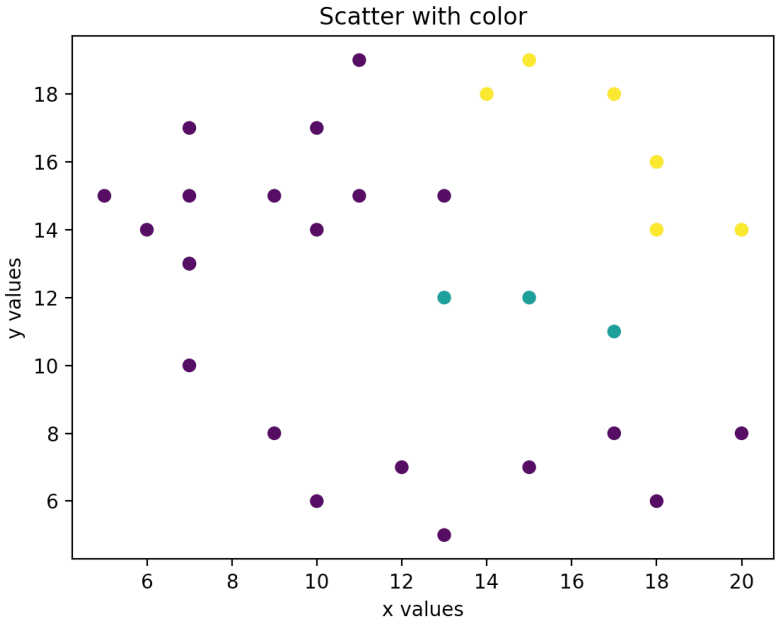
1. yuq8@andromeda-23:~/253P/hw_lab/8/1_KCluster (ssh)

..P_APPS/hw_lab (zsh) ⌘1 | yuq8@andromeda-2... ⌘2 | ..lassification (zsh) ⌘3 | ~ (zsh) ⌘4

```
$ valgrind ./main
==24641== Memcheck, a memory error detector
==24641== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==24641== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==24641== Command: ./main
==24641==
1 13 15
2 7 10
3 15 12
4 13 12
5 7 15
6 18 16
7 10 6
8 9 15
9 10 17
10 7 13
11 15 19
12 10 14
13 17 8
14 12 7
15 18 14
16 14 18
17 7 13
18 11 19
19 7 17
20 18 6
21 20 14
22 20 8
23 9 8
24 5 15
25 17 18
26 11 15
27 15 7
28 17 11
29 13 5
30 6 14
Read data successfully!
InitialDistanceTable done.
```

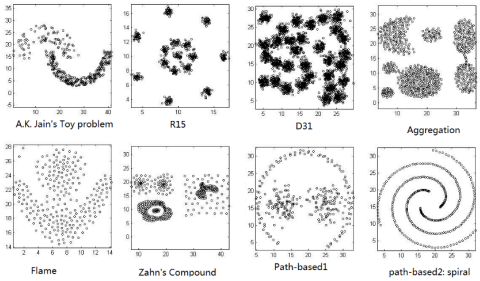
```
count cluster times: 26
FindClosestCluster done.
MergeMJtoMI done.
RemoveMJinDataset done.
UpdateDistanceTable done.
RemoveMJindTable done.

count cluster times: 27
FindClosestCluster done.
MergeMJtoMI done.
RemoveMJinDataset done.
UpdateDistanceTable done.
RemoveMJindTable done.

n_clusters in result: 3 among 30
==24641==
==24641== HEAP SUMMARY:
==24641==     in use at exit: 72,704 bytes in 1 blocks
==24641==   total heap usage: 1,242 allocs, 1,241 frees, 116,956 bytes allocated
==24641==
==24641== LEAK SUMMARY:
==24641==    definitely lost: 0 bytes in 0 blocks
==24641==    indirectly lost: 0 bytes in 0 blocks
==24641==      possibly lost: 0 bytes in 0 blocks
==24641==    still reachable: 72,704 bytes in 1 blocks
==24641==         suppressed: 0 bytes in 0 blocks
==24641== Rerun with --leak-check=full to see details of leaked memory
==24641==
==24641== For counts of detected and suppressed errors, rerun with: -v
==24641== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
(base) yuq8@andromeda-23 13:59:09 ~/253P/hw_lab/8/1_KCluster
$
```





Stepping further, I find some test set in : http://cs.joensuu.fi/sipu/datasets/ , for example:

Shape sets



Third column is the label.

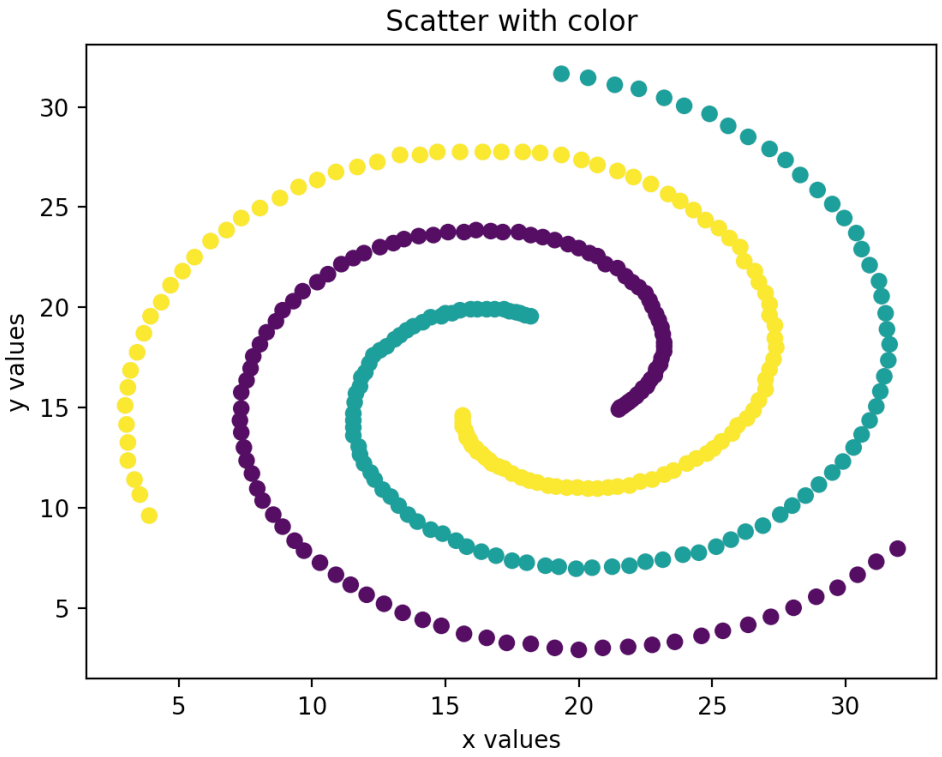
Aggregation N=788, k=7, D=2	**Aggregation:** txt A. Gionis, H. Mannila, and P. Tsaparas, Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2007. 1(1): p. 1-30
Compound N=399, k=6, D=2	**Compound:** txt C.T. Zahn, Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, 1971. 100(1): p. 68-86.
Pathbased N=300, k=3, D=2	**Pathbased:** txt H. Chang and D.Y. Yeung, Robust path-based spectral clustering. *Pattern Recognition*, 2008. 41(1): p. 191-203.
Spiral N=312, k=3, D=2	**Spiral:** txt H. Chang and D.Y. Yeung, Robust path-based spectral clustering. *Pattern Recognition*, 2008. 41(1): p. 191-203.
D31 N=3100, k=31, D=2	**D31:** txt C.J. Veenman, M.J.T. Reinders, and E. Backer, A maximum variance cluster algorithm. *IEEE Trans. Pattern Analysis and Machine Intelligence* 2002. 24(9): p. 1273-1280.
R15 N=600, k=15, D=2	**R15:** txt C.J. Veenman, M.J.T. Reinders, and E. Backer, A maximum variance cluster algorithm. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2002. 24(9): p. 1273-1280.
Jain N=373, k=2, D=2	**Jain:** txt A. Jain and M. Law, Data clustering: A user's dilemma. *Lecture Notes in Computer Science*, 2005. 3776: p. 1-10.
Flame N=240, k=2, D=2	**Flame:** txt L. Fu and E. Medico, FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data. *BMC bioinformatics*, 2007. 8(1): p. 3.

and I transferred them to the desires format as our std input:

main.cpp — 8/1_KCluster × pretransformat.cpp × input.txt

```cpp
#include <iostream>
#include <cstdio>
#include <vector>
#include <fstream>
#include <cmath>
#include <utility>
using namespace std;

int main() {
    string infile = "spiral.txt";
    int N=312, k=3;
    fstream in;
    in.open(infile);

    fstream outfile;
    //outfile.open("clustering.txt", ios::out);
    outfile.open("spiral_format.txt", ios::out);
    outfile<<N<<endl;
    outfile<<k<<endl;

    int count = 0;
    while( !in.eof() ){
        int i;
        float x, y;
        in >> x >> y >> i;
        cout<<x<<" "<<y<<" "<<i<<" "<<endl;
        outfile<<count+1<<" "<< x <<" " << y <<endl;
        count++;
    }
    in.close();
    cout<<"Read data successfully!"<<endl;

    return 0;
}
```

and the testcase 3 result are as follow:



Conclusion: right result, 0 error in memory leak.
Little Notes:

1) The different clustering critierion will make different results, like shortest, mean, longest.

Lab 2: closest product pair

Code:

```cpp
void MyQuickSort(vector<int>& arr, int low, int high){
    if (low >= high) return; //
    int pivot = arr[high]; //pivot
    int i = low -1; //index of the less/equal position
    for (int j = low; j < high; j++){ //j = low!!!
        if (arr[j] <= pivot){
            i++;
            swap(arr[i], arr[j]);//Finally place pivot at correct position by swapping arr[i+1] and arr[high] (or pivot)
        }
        //cout<<arr.size()<<endl;
        //PrintVector(arr, arr.size());
    }
    swap(arr[++i], arr[high]);
    //PrintVector(arr, arr.size());
    //cout<<"one time recursive: i = "<<i<<endl;
    MyQuickSort(arr, low, i - 1);
    MyQuickSort(arr, i + 1, high);
}
int main() {
    //int a[] = {2,3,5,9};
    //int a[] = {9,5,3,2};
    //int a[] = {4,6,8,10,20,30};
    int a[] = {4,20,8,10,6,30};
    //int target = 8;
    //int target = 47;
    int target = 268;

    int len = sizeof(a) / sizeof(a[0]);
    //cout<<len<<endl;
    vector<int> arr(a,a+len);
    cout<<"input string: "<<endl;
    PrintVector(arr, len);
    //cout<<arr.size()<<endl;
    MyQuickSort(arr, 0, len-1);
    cout<<"finish sort, string is: "<<endl;
    PrintVector(arr, len);
    CloestProductPair(arr, target, len);
    return 0;
}
```

Test cases 1, 2, 3 :

```
$ cd 2_ClosestProduct/
(base) yuq8@andromeda-23 14:04:24 ~/253P/hw_lab/8/2_ClosestProduct
$ ls
1.log  main*  main.cpp  main.dSYM/  Makefile
(base) yuq8@andromeda-23 14:04:25 ~/253P/hw_lab/8/2_ClosestProduct
$ make
------------compiling main.cpp to create executable program main---------------
g++  -ggdb   -std=c++11   main.cpp   -o   main
------------Congratulation to you! Successfully compile.
------------Run manually by :
------------./main
(base) yuq8@andromeda-23 14:04:27 ~/253P/hw_lab/8/2_ClosestProduct
```

```
$ valgrind ./main
==28903== Memcheck, a memory error detector
==28903== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==28903== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==28903== Command: ./main
==28903==
input string:
4  20  8  10  6  30
finish sort, string is:
4  6  8  10  20  30
 The target is 268
 The closest pair is 8 and 30
==28903==
==28903== HEAP SUMMARY:
==28903==     in use at exit: 72,704 bytes in 1 blocks
==28903==   total heap usage: 6 allocs, 5 frees, 72,824 bytes allocated
==28903==
==28903== LEAK SUMMARY:
==28903==    definitely lost: 0 bytes in 0 blocks
==28903==    indirectly lost: 0 bytes in 0 blocks
==28903==      possibly lost: 0 bytes in 0 blocks
==28903==    still reachable: 72,704 bytes in 1 blocks
==28903==         suppressed: 0 bytes in 0 blocks
==28903== Rerun with --leak-check=full to see details of leaked memory
==28903==
==28903== For counts of detected and suppressed errors, rerun with: -v
==28903== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
(base) yuq8@andromeda-23 14:04:58 ~/253P/hw_lab/8/2_ClosestProduct
$
```

```
$ make
------------compiling main.cpp to create executable program main----------------
g++  -ggdb   -std=c++11   main.cpp   -o   main
------------Congratulation to you! Successfully compile.
------------Run manually by :
------------./main
(base) yuq8@andromeda-23 14:05:56 ~/253P/hw_lab/8/2_ClosestProduct
$ valgrind ./main
==29177== Memcheck, a memory error detector
==29177== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==29177== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==29177== Command: ./main
==29177==
input string:
9  5  3  2
finish sort, string is:
2  3  5  9
 The target is 47
 The closest pair is 5 and 9
==29177==
==29177== HEAP SUMMARY:
==29177==     in use at exit: 72,704 bytes in 1 blocks
==29177==   total heap usage: 6 allocs, 5 frees, 72,784 bytes allocated
==29177==
==29177== LEAK SUMMARY:
==29177==    definitely lost: 0 bytes in 0 blocks
==29177==    indirectly lost: 0 bytes in 0 blocks
==29177==      possibly lost: 0 bytes in 0 blocks
==29177==    still reachable: 72,704 bytes in 1 blocks
==29177==         suppressed: 0 bytes in 0 blocks
==29177== Rerun with --leak-check=full to see details of leaked memory
==29177==
==29177== For counts of detected and suppressed errors, rerun with: -v
==29177== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
(base) yuq8@andromeda-23 14:06:34 ~/253P/hw_lab/8/2_ClosestProduct
$
```

```
$ make
------------compiling main.cpp to create executable program main----------------
g++  -ggdb   -std=c++11   main.cpp   -o   main
------------Congratulation to you! Successfully compile.
------------Run manually by :
------------./main
(base) yuq8@andromeda-23 14:07:08 ~/253P/hw_lab/8/2_ClosestProduct
$ valgrind ./main
==29329== Memcheck, a memory error detector
==29329== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==29329== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==29329== Command: ./main
==29329==
input string:
2  3  5  9
finish sort, string is:
2  3  5  9
 The target is 8
 The closest pair is 2 and 5
==29329==
==29329== HEAP SUMMARY:
==29329==     in use at exit: 72,704 bytes in 1 blocks
==29329==   total heap usage: 6 allocs, 5 frees, 72,784 bytes allocated
==29329==
==29329== LEAK SUMMARY:
==29329==    definitely lost: 0 bytes in 0 blocks
==29329==    indirectly lost: 0 bytes in 0 blocks
==29329==      possibly lost: 0 bytes in 0 blocks
==29329==    still reachable: 72,704 bytes in 1 blocks
==29329==         suppressed: 0 bytes in 0 blocks
==29329== Rerun with --leak-check=full to see details of leaked memory
==29329==
==29329== For counts of detected and suppressed errors, rerun with: -v
==29329== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
(base) yuq8@andromeda-23 14:07:12 ~/253P/hw_lab/8/2_ClosestProduct
$
```

Conclusion: right result, 0 error in memory leak.
Little Notes:

1) At this lab, when encountered with the same diff distance, the results are given by larger distance. Furthur work can be done to output both.

Lab4:Longest K

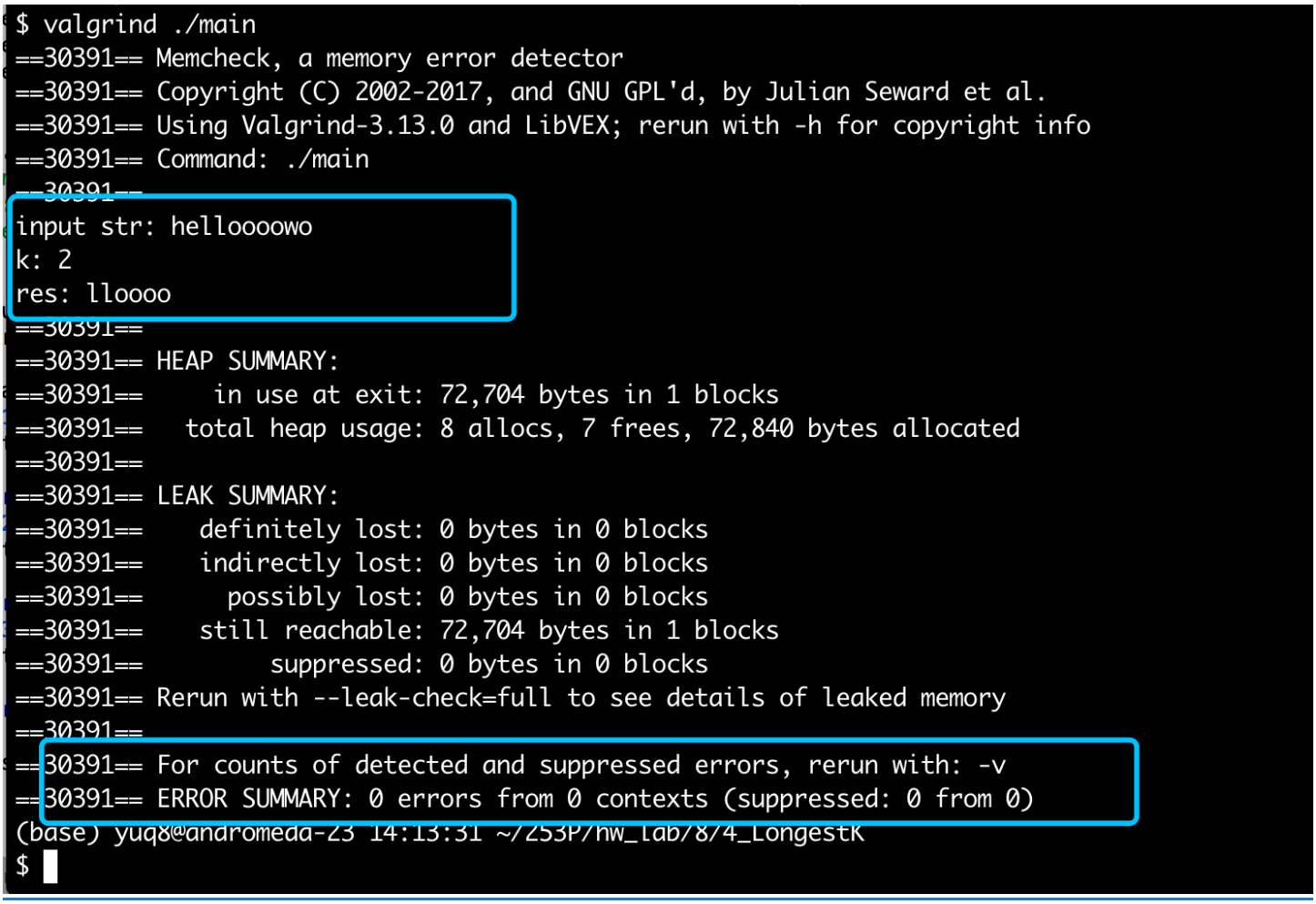
Code：

```
7
8   void Longestk(string str, int k){
9       int len = str.size();
10      string res;
11      int reslen = 0, res_l = 0;
12      unordered_map<char, int> m;
13      int left = 0;
14      for (int i = 0; i < len; i++){
15          ++m[str[i]];
16          while ( m.size() > k ){
17              if ( --m[str[left]] == 0)
18                  m.erase(str[left]);
19              ++left;
20          }
21          if (reslen < i - left + 1){
22              reslen = i - left +1;
23              res_l = left;
24          }
25      }
26      res = str.substr(res_l, reslen);
27      cout <<"input str: "<<str<<endl;
28      cout <<"k: "<<k<<endl;
29      cout <<"res: "<<res<<endl;
30  }
31  int main() {
32      int casenumber = 1;
33      string str;
34      int k;
35      switch (casenumber){
36          case 1 :
37              str = "abcbbbbcccbdddadacb";
38              k = 2;
39              break;
40          case 2 :
41              str = "abcbbbbcccbdddadacb";
42              k = 3;
43              break;
44          case 3 :
45              str = "hellooooowo";
46              k = 2;
47              break;
48      }
49      Longestk(str, k);
50      return 0;
51  }
```

test cases 1,2, 3:

```
$ valgrind ./main
==30171== Memcheck, a memory error detector
==30171== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==30171== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==30171== Command: ./main
==30171==
input str: abcbbbbcccbdddadacb
k: 2
res: bcbbbbcccb
==30171==
==30171== HEAP SUMMARY:
==30171==     in use at exit: 72,704 bytes in 1 blocks
==30171==   total heap usage: 12 allocs, 11 frees, 72,923 bytes allocated
==30171==
==30171== LEAK SUMMARY:
==30171==    definitely lost: 0 bytes in 0 blocks
==30171==    indirectly lost: 0 bytes in 0 blocks
==30171==      possibly lost: 0 bytes in 0 blocks
==30171==    still reachable: 72,704 bytes in 1 blocks
==30171==         suppressed: 0 bytes in 0 blocks
==30171== Rerun with --leak-check=full to see details of leaked memory
==30171==
==30171== For counts of detected and suppressed errors, rerun with: -v
==30171== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
(base) yuq8@andromeda-23 14:12:11 ~/253P/hw_lab/8/4_LongestK
$
```

```
$ vi main.cpp
(base) yuq8@andromeda-23 14:12:44 ~/253P/hw_lab/8/4_LongestK
$ make
------------compiling main.cpp to create executable program main----------------
g++  -ggdb   -std=c++11   main.cpp   -o   main
------------Congratulation to you! Successfully compile.
------------Run manually by :
------------./main
(base) yuq8@andromeda-23 14:12:47 ~/253P/hw_lab/8/4_LongestK
$ valgrind ./main
==30281== Memcheck, a memory error detector
==30281== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==30281== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==30281== Command: ./main
==30281==
input str: abcbbbbcccbdddadacb
k: 3
res: bcbbbbcccbddd
==30281==
==30281== HEAP SUMMARY:
==30281==     in use at exit: 72,704 bytes in 1 blocks
==30281==   total heap usage: 12 allocs, 11 frees, 72,923 bytes allocated
==30281==
==30281== LEAK SUMMARY:
==30281==    definitely lost: 0 bytes in 0 blocks
==30281==    indirectly lost: 0 bytes in 0 blocks
==30281==      possibly lost: 0 bytes in 0 blocks
==30281==    still reachable: 72,704 bytes in 1 blocks
==30281==         suppressed: 0 bytes in 0 blocks
==30281== Rerun with --leak-check=full to see details of leaked memory
==30281==
==30281== For counts of detected and suppressed errors, rerun with: -v
==30281== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
(base) yuq8@andromeda-23 14:12:50 ~/253P/hw_lab/8/4_LongestK
$
```


$ valgrind ./main
==30391== Memcheck, a memory error detector
==30391== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==30391== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==30391== Command: ./main
==30391==
input str: helloooowo
k: 2
res: lloooo
==30391==
==30391== HEAP SUMMARY:
==30391==   in use at exit: 72,704 bytes in 1 blocks
==30391==   total heap usage: 8 allocs, 7 frees, 72,840 bytes allocated
==30391==
==30391== LEAK SUMMARY:
==30391==   definitely lost: 0 bytes in 0 blocks
==30391==   indirectly lost: 0 bytes in 0 blocks
==30391==   possibly lost: 0 bytes in 0 blocks
==30391==   still reachable: 72,704 bytes in 1 blocks
==30391==   suppressed: 0 bytes in 0 blocks
==30391== Rerun with --leak-check=full to see details of leaked memory
==30391==
==30391== For counts of detected and suppressed errors, rerun with: -v
==30391== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
(base) yuq8@andromeda-23 14:13:31 ~/253P/hw_lab/8/4_LongestK
$