```c
/* MaxumumGap.c */
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <malloc.h>
#include <time.h>

static int IntHash(int key, int lo, int hi, int n)
{
    return (n - 1) * (key - lo) / (hi - lo);
}

static void swap(int *a, int *b)
{
    int tmp = *a;
    *a = *b;
    *b = tmp;
}

int main()
{
    time_t t;
    //初始化随机数发生器
    srand((unsigned)time(&t));
    int points[100];
    int i = 0;
    int min = 900001, max = 0;
    for (; i < 100; i ++)
    {
        points[i] = rand() % 900001;
        if (min > points[i])
        {
            min = points[i];
        }
        if (max < points[i])
        {
            max = points[i];
        }
        if (i % 10 == 9)
        {
            int j = i - 9;
            for (; j <= i; j ++)
            {
                printf("%6d  ", points[j]);
            }
            printf("\n");
        }
    }
    int hashTable[100][2];
    memset(hashTable, -1, 100 * 2 * sizeof(int));
    for (i = 0; i < 100; i ++)
    {
        int hashKey = IntHash(points[i], min, max, 100);
        if (-1 == hashTable[hashKey][0])
        {
            hashTable[hashKey][0] = points[i];
            continue;
        }
        if (-1 == hashTable[hashKey][1])
        {
            hashTable[hashKey][1] = points[i];
            if (hashTable[hashKey][0] > hashTable[hashKey][1])
            {
                swap(&(hashTable[hashKey][0]), &(hashTable[hashKey][1]));
            }
            continue;
        }
        if (points[i] < hashTable[hashKey][0])
        {
            hashTable[hashKey][0] = points[i];
            continue;
        }
        if (points[i] > hashTable[hashKey][1])
```

```
74              {
75                  hashTable[hashKey][1] = points[i];
76              }
77          }
78          int head, tail;
79          int maximumGap = 0;
80          int rightVal = min;
81          int gap;
82          for (i = 0; i < 100; i ++)
83          {
84              if (-1 == hashTable[i][0])
85              {
86                  continue;
87              }
88              gap = hashTable[i][0] - rightVal;
89              if (maximumGap < gap)
90              {
91                  maximumGap = gap;
92                  head = rightVal;
93                  tail = hashTable[i][0];
94              }
95              if (-1 == hashTable[i][1])
96              {
97                  rightVal = hashTable[i][0];
98              }
99              else
100             {
101                 rightVal = hashTable[i][1];
102             }
103         }
104         printf("maximum gap is %d, range head = %d, range tail = %d\n", maximumGap,
            head, tail);
105         return 0;
106     }
```