
BARCODE

راهنمای تیم بارکد
همراه با کدهای آماده برای مسابقات

تیم بارکد (Barcode)

۲۵ دی ۱۳۹۹

فهرست مطالب

مقدمه

فصل ۱

کدهای آماده (Cheat Sheets)

C++ ۱.۱

dijkstra ۱.۱.۱

لینک برگرفته شده از سایت geeks for geeks الگوریتم Dijkstra's shortest path algorithm. [۱]

```
#include <climits>
#define V 9

int minDistance(int dist[], bool sptSet[]) {
    int min = INT_MAX, min_index;
    for (int v = 0; v < V; v++)
    if (sptSet[v] == false && dist[v] <= min)
    min = dist[v], min_index = v;
    return min_index;
}

void dijkstra(int graph[V][V], int src) {
    int dist[V];
    bool sptSet[V];
    for (int i = 0; i < V; i++)
    dist[i] = INT_MAX, sptSet[i] = false;
    dist[src] = 0;
    for (int count = 0; count < V - 1; count++) {
        int u = minDistance(dist, sptSet);
        sptSet[u] = true;
        for (int v = 0; v < V; v++)
        if (!sptSet[v] && graph[u][v] && dist[u] != INT_MAX &&
```

```
23                              dist[u] + graph[u][v] < dist[v])
24            dist[v] = dist[u] + graph[u][v];
25    }
26    printSolution(dist);
27 }
```

segment tree ۲.۱.۱

لینک برگرفته شده از سایت geeks for geeks الگوریتم Dijkstra's shortest path algorithm. [۱]

```
1  const int N = 100000;
2  int n;
3  int tree[2 * N];
4
5  void build( int arr[]) {
6     for (int i=0; i<n; i++)
7        tree[n+i] = arr[i];
8     for (int i = n - 1; i > 0; --i)
9        tree[i] = tree[i<<1] + tree[i<<1 | 1];
10 }
11
12 void updateTreeNode(int p, int value) {
13    tree[p+n] = value;
14    p = p+n;
15    for (int i=p; i > 1; i >>= 1)
16       tree[i>>1] = tree[i] + tree[i^1];
17 }
18
19 int query(int l, int r) {
20    int res = 0;
21    for (l += n, r += n; l < r; l >>= 1, r >>= 1) {
22       if (l&1)
23          res += tree[l++];
24       if (r&1)
25          res += tree[--r];
26    }
27    return res;
28 }
```

big int multiplication ۳.۱.۱

revise needed...

```cpp
#include <string>
#include <algorithm>

tring multiplication(string str1, string str2) {
   int len1 = str1.length(), len2 = 0, olen = 0;
   string res(len1 + len2, 0);
   for (int i = 0; i <len2; i++) {
      for (int j = 0; j <len1; j++) {
         res[j + i] += (str1[len1 - j - 1] - 48) * (str2[len2 - i - 1]
            ↪ - 48);
         olen = j + i + 1;
         if (res[j + i] >= 10) {
            res[j + i + 1] += res[j + i] / 10;
            res[j + i] = res[j + i] % 10;
            olen = j + i + 2;
         }
      }
   }
   string reverseOut = reverse(res.begin(),res.end());
   return reverseOut;
```

K-D tree ۴.۱.۱

لینک برگرفته شده از سایت geeks for geeks الگوریتم kd-tree. [۱]

```cpp
#include<bits/stdc++.h>
using namespace std;

const int k = 2;

struct Node {
    int point[k];
    Node *left, *right;
};

struct Node* newNode(int arr[]) {
    struct Node* temp = new Node;
    for (int i=0; i<k; i++)
        temp->point[i] = arr[i];
```

```
15      temp->left = temp->right = NULL;
16      return temp;
17  }
18
19  Node *insertRec(Node *root, int point[], unsigned depth) {
20      if (root == NULL)
21          return newNode(point);
22      unsigned cd = depth % k;
23      if (point[cd] < (root->point[cd]))
24          root->left = insertRec(root->left, point, depth + 1);
25      else
26          root->right = insertRec(root->right, point, depth + 1);
27      return root;
28  }
29
30  Node* insert(Node *root, int point[]) {
31      return insertRec(root, point, 0);
32  }
33
34  bool arePointsSame(int point1[], int point2[]) {
35      for (int i = 0; i < k; ++i)
36          if (point1[i] != point2[i])
37              return false;
38      return true;
39  }
40
41  bool searchRec(Node* root, int point[], unsigned depth) {
42      if (root == NULL)
43          return false;
44      if (arePointsSame(root->point, point))
45          return true;
46
47      unsigned cd = depth % k;
48      if (point[cd] < root->point[cd])
49          return searchRec(root->left, point, depth + 1);
50      return searchRec(root->right, point, depth + 1);
51  }
52
53  bool search(Node* root, int point[]) {
54      return searchRec(root, point, 0);
55  }
56
57  int main() {
58      struct Node *root = NULL;
59      int points[][k] = {{3, 6}, {17, 15}, {13, 15}, {6, 12}, {9, 1}, {2,
```

```
        ↪ 7}, {10, 19}};
60
61    int n = sizeof(points)/sizeof(points[0]);
62
63    for (int i=0; i<n; i++)
64       root = insert(root, points[i]);
65
66    int point1[] = {10, 19};
67    (search(root, point1))? cout << "Found\n": cout << "Not Found\n";
68
69    int point2[] = {12, 19};
70    (search(root, point2))? cout << "Found\n": cout << "Not Found\n";
71
72    return 0;
73 }
```

Longest Common Subsequence ۵.۱.۱

لینک برگرفته شده از سایت geeks for geeks الگوریتم Longest Common Subsequence. [۱]

```
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  int max(int a, int b);
5
6  int lcs( char *X, char *Y, int m, int n ) {
7     int L[m + 1][n + 1];
8     int i, j;
9     for (i = 0; i <= m; i++) {
10       for (j = 0; j <= n; j++) {
11          if (i == 0 || j == 0)
12             L[i][j] = 0;
13          else if (X[i - 1] == Y[j - 1])
14             L[i][j] = L[i - 1][j - 1] + 1;
15          else
16             L[i][j] = max(L[i - 1][j], L[i][j - 1]);
17       }
18    }
19    return L[m][n];
20 }
21
22 int max(int a, int b) {
23    return (a > b)? a : b;
```

```
}

int main() {
    char X[] = "AGGTAB";
    char Y[] = "GXTXAYB";

    int m = strlen(X);
    int n = strlen(Y);

    cout << "Length of LCS is " << lcs( X, Y, m, n );

    return 0;
}
```

algorithm name ۶.۱.۱

Python ۲.۱

algorithm name ۱.۲.۱

Java ۳.۱

algorithm name ۱.۳.۱

فصل ۲

قوانین مسابقه

فصل ۳

توضیحات الگوریتم‌ها

مراجع

[1] Geeksforgeeks | a computer science portal for geeks. https://www.geeksforgeeks.org/. (Accessed on 12/27/2020).

[2] cppreference.com. https://en.cppreference.com/w/. (Accessed on 12/27/2020).

[3] Cormen, Thomas H., Leiserson, Charles E., Rivest, Ronald L., and Stein, Clifford. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd ed. , 2009.