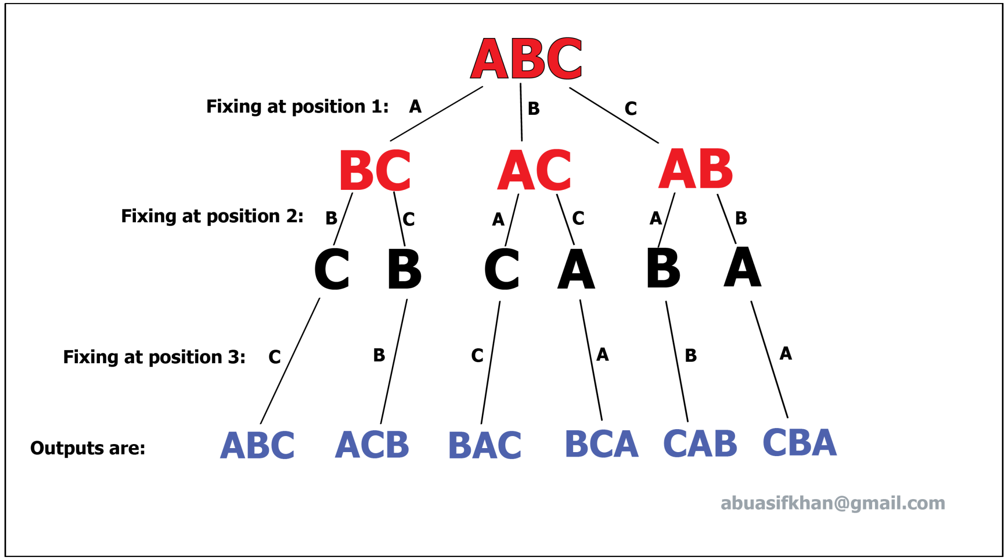
বিন্যাস করা যাক (পর্ব: ২)

October 26th, 2013

গতপর্বে permutation করার বেসিক জিনিসগুলা দেখানোর চেষ্টা করেছিলাম। আজকের পর্বে শেয়ার করবো কিভাবে permutation generator কোড করা যায়। কোডিং করার আগে তো আমাদের অবশ্যই জানতে হবে কিভাবে permutation generate করা হয়। আগে সেটা দেখাই।

নিজে কিভাবে করবে?



উপরের ছবিটার মত ধরে নাও “ABC” এই তিনটা character গুলাকে আমরা বিন্যাস করতে চাই। তো এই বিশাল কর্ম সম্পাদনের জন্য আমাদেরকে প্রথমে বিভিন্ন পজিশনে বর্ণগুলাকে fixed করে রেখে পরের পজিশনের কথা বিবেচনা করতে হবে। যদি একদম শেষ পজিশনে এসে যাই তাইলে প্রথম থেকে শেষ পর্যন্ত fixed করে রাখা পজিশনগুলার সবগুলা character একসাথে প্রিন্ট করে দিবো। যেমন: প্রথমে আমরা 1 no. পজিশনে 'A' fixed রেখেছি, তারপর 2 no. পজিশনে 'B', তারপর 3 no. পজিশনে রেখেছি 'C'. সুতরাং প্রথম বিন্যাস আমরা পাচ্ছি “ABC”. এভাবে পরে 2 no. পজিশনে 'C' fixed করেছি। তাইলে 3 no. পজিশনে কেবল আমরা 'B' কেই বসাতে পারি। তাই পরের বিন্যাসটা আমরা পাচ্ছি “ACB”. এভাবে ট্রীতে আস্তে আস্তে উপরে লেভেলে যেয়ে যেয়ে এবং বিভিন্ন

  - খাতা-কলমে Extended Euclid Method
  - Extended Euclidean Algorithm এবং একটুখানি Modular Multiplicative Inverse

- Recent Comments
  - Muhammad Minhazul Haque on বিন্যাস করা যাক (পর্ব: ২)
  - Duronto Habib on বিন্যাস করা যাক (পর্ব: ১)
  - Abu Asif Khan Chowdhury on Chinese Remainder Theorem
  - TripleM Zim on Chinese Remainder Theorem
  - Abu Asif Khan Chowdhury on Chinese Remainder Theorem

- Blog Traffic

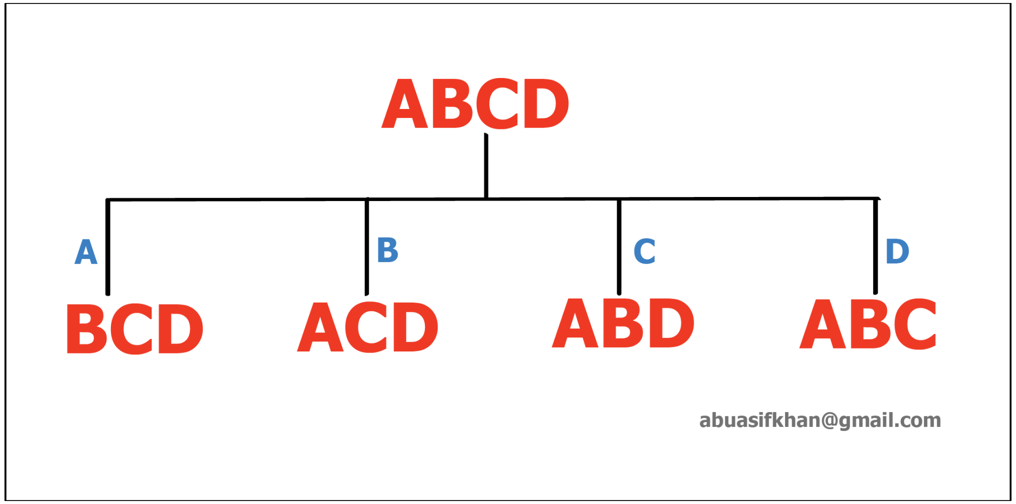
Pages

Pages | Hits | Unique

  - Last 24 hours: 147
  - Last 7 days: 472

পজিশনে বিভিন্ন character বসিয়ে আমরা সকল সাম্ভাব্য বিন্যাস generate করতে পারবো।

যদি 4 টা character দেয়া থাকে বা তার বেশি? তখন কিভাবে করবে? উপরে যেভাবে করেছো সেভাবেই করে যাবে। আচ্ছা জিনিসটা সহজ করার জন্য নিচে “ABCD” characterগুলা থেকে সকল বিন্যাস গঠন করার প্রথম ধাপটা দিলাম।



1 no. পজিশনে প্রথমে ‘A’ fixed রেখে বাকি characterগুলা অর্থাৎ “BCD” এর বিন্যাসগুলা প্রথম চিত্রের মত করে করে ফেলো। তারপর প্রথম পজিশনে ‘B’ কে fixed রেখে “ACD” characterগুলা আগের মত উপায়ে বিন্যাস করে ফেলো। এভাবে বাকি দুইটা করে ফেললে তুমি মোট 4! = 24 টা সাম্ভাব্য বিন্যাস গঠন করতে পারবে। কাজ সহজ, খাটনি একটু এই আর কি। :P

  - Last 30 days: 758
  - Online now: 1

- Get Updates

  Join 1 other subscriber

  Email Address

  Subscribe

- Meta
  - Log in
  - Entries RSS
  - Comments RSS
  - WordPress.org

প্রোগ্রামে কিভাবে করবে?

আচ্ছা এখন million dollar question হলো কিভাবে তুমি এটা প্রোগ্রামে করতে পারবে? যদি একটা সাধারন প্রোগ্রামারকে বলা হয় যে তিন এলিমেন্টের কোন অ্যারের বিন্যাসগুলা খুজতে সে খুব সহজেই ৩ টা for loop ব্যবহার করে সেটা করে ফেলতে পারবে। কিন্তু যদি বলি ৪, ৫, ১০, ২০? তখন কি প্রতিবার অতগুলা লুপ সে তৈরী করতে পারবে? নাহ, এভাবে সব কার্যদ্ধার করা যাবে না। এজন্য আমাদের লাগবে রিকার্সিভ ব্যাকট্রাকিং। এই টেকনিকের বড় সুবিধা হলো আমরা সম্পুর্ন search spaceটা search করতে পারি অত লুপ-টুপ ব্যবহার না করেই। তবে সমস্যাও আছে, search space অনেক বড় হলে এই পদ্ধতি খুব বেশি সময় অপচয়কারী হয়ে দাড়ায়। এক্সপোনেন্সিয়াল আকারে বাড়তে থাকে বলে খুব বড় ইনপুটের ক্ষেত্রে সমাধান বের করতে কয়েক বছর লাগিয়ে দিতে পারে, যেটা কনটেস্টের কোন প্রবলেমের সমাধান হিসেবে জাজদের মনমত হওয়ার সুযোগ কম, TLE খাওয়া লাগতে পারে। তবে সকল সাম্ভব্য বিন্যাস generate করতে এটাই ব্যবহার করা হয়। এখন কাজের কথায় আসি, আমরা all possible permutation generate করার জন্য একটা ফাংশন তৈরী করবো ব্যাকট্রাকিং এর সাহায্যে যেটাতে আমরা উপরের যে পদ্ধতি বিন্যাস করার জন্য ব্যবহার করেছি সেটাই করবো। নিচে কোডটা দিলাম, ওখানে ব্যাখ্যাও দেয়া আছে একটু-অর্ধেক...

C++

```cpp
#include <iostream>
#include <cstdio>
#include <vector>

using namespace std;

vector<char>permuted;
int fixedPos[20]= {0};
```

```

///ব্যাকট্রাক মেথড
void generate(int array[]){
    if(permuted.size()==4){    /// যদি সব কয়টা পজিশন ফিক্সড হয়ে যায়
        for(int i=0; i<4; i++)
            printf("%c",permuted[i]); ///তাইলে প্রিন্ট করতে হবে permuted
        printf("\n");
        return;
    }
    for(int i=0; i<4; i++){
        if(fixedPos[i]==0){    /// যদি অ্যারের i-তম পজিশনের characterটা
            fixedPos[i]=1;         /// তাইলে fixed করে নিলাম
            permuted.push_back(array[i]);    /// ফিক্সড character টা
            generate(array);    /// ট্রীতে নিচের লেভেলে যাওয়ার জন্য রিকার্সিভ
            fixedPos[i]=0;         /// i-তম পজিশনে array[i] characte
            permuted.pop_back();    /// কাজ শেষ তো character টা তো
        }
    }
}
int main(){
    int array[] = {'A', 'B', 'C', 'D'};
    generate(array);
}
```

আপাতত এ পর্যন্তই। পরবর্তি পর্বে n-তম বিন্যাস কিভাবে বের করা যায় সেটা দেখানোর চেষ্টা করবো। আরও ভালো জানতে শাফায়েত ভাইয়ের লেখা আর্টিকেলটা পড়তে পারো। আর যা যা শিখলে সেগুলা প্রাকটিস করার জন্য কিছু প্রবলেম দিলাম। Try to solve them:

524 - Prime Ring Problem

10776 - Determine The Combination

291 - The House of Santa Claus

677 - All walks of length n from the first node

Keep coding… :)

75 total views, 32 views today

Share this:

Posted in Uncategorized | 1 Comment »

- *http://minhazulhaque.com/ Muhammad Minhazul Haque*

  অনেক ভাল লাগলো ভাইয়া্ অসংখ্য ধন্যবাদ রিকারসিভ সলুশনটার জন্য।