1-5 Data Structures

魏恒峰

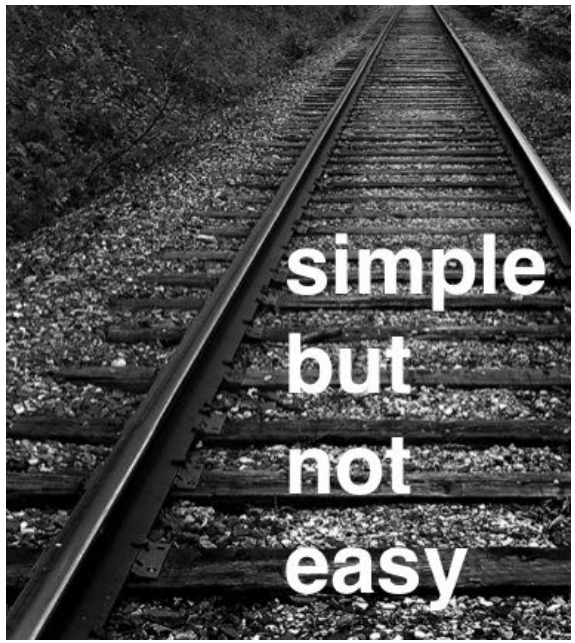
hfwei@nju.edu.cn

2019 年 11 月 07 日

Permutations

Generating All Permutations
Stackable/Queueable Permutations

Generating All Permutations



DH 2.11: Generate All Permutations

Design an algorithm which, given a positive integer n, generates/prints all the permutations of $[0 \cdots n)$.

DH 2.11: Generate All Permutations

Design an algorithm which, given a positive integer n, generates/prints all the permutations of $[0 \cdots n)$.

```
void perms (A[], n) {
  if (n == 1)
    print ''A[0]''
  else
    for (int i = 0; i < n; ++i)
      print ''A[i]''
      perms(A ← A \ A[i], n - 1)
      print ''\n''
}
```

DH 2.11: Generate All Permutations

Design an algorithm which, given a positive integer n, generates/prints all the permutations of $[0 \cdots n)$.

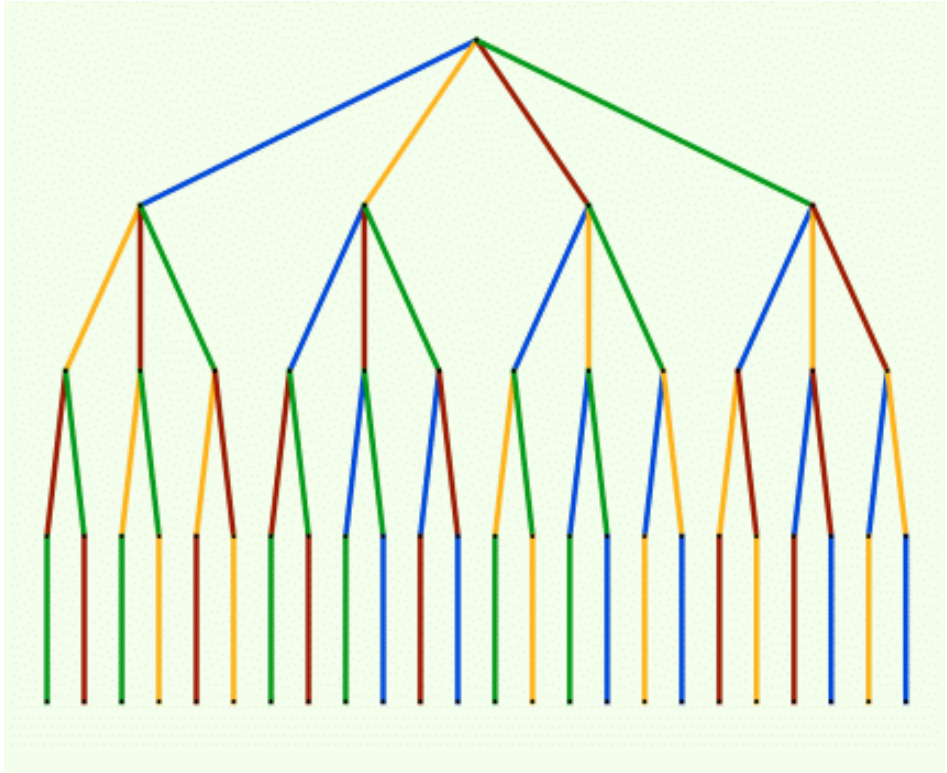
```
void perms (A[], n) {
  if (n == 1)
    print ''A[0]''
  else
    for (int i = 0; i < n; ++i)
      print ''A[i]''
      perms(A ← A \ A[i], n - 1)
      print ''\n''
}
```

generate-perms.c

4perms.md

4perms.md

$$A = [0, 1, 2, 3] \qquad n = 4$$



“手动单步调试”

```
void perms (prifix, A[], n) {
  if (n == 1)
    print ''prifix ++ A[0]''
  else
    for (int i = 0; i < n; ++i)
      perms(prefix ← prefix ++ A[i],
          A ← A \ A[i], n - 1)
      print ''\n''
}
```

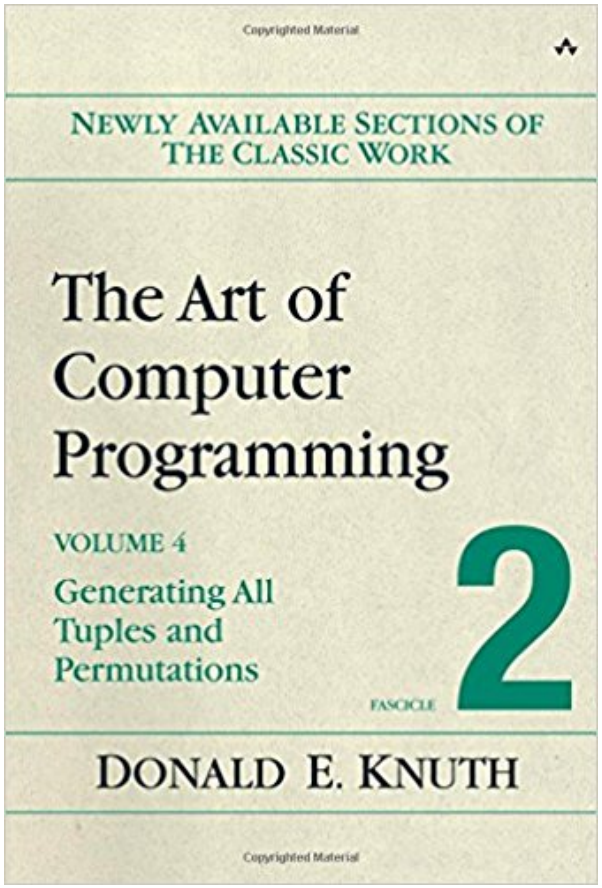
```
void perms (prifix, A[], n) {
  if (n == 1)
    print ''prifix ++ A[0]''
  else
    for (int i = 0; i < n; ++i)
      perms(prefix ← prefix ++ A[i],
          A ← A \ A[i], n - 1)
      print ''\n''
}
```

```
perms('''', A, n);
```

```
void perms (prifix, A[], n) {
  if (n == 1)
    print ''prifix ++ A[0]''
  else
    for (int i = 0; i < n; ++i)
      perms(prefix ← prefix ++ A[i],
          A ← A \ A[i], n - 1) // Space???
      print ''\n''
}
```

```
perms('''', A, n);
```

For more about "Generating All Permutations":



Thank
You!