C 上机

函数

张晓平

武汉大学数学与统计学院

homepage: xpzhang.me

2018 年 6 月 3 日

目录

编程

设计函数$min(x, y)$，返回两个`double`数值中的较小值，并测试之。

```c
// min.c
#include <stdio.h>
#include <stdlib.h>

double min(double x, double y);

int main(int argc, char * argv[])
{
  double x, y;
  if(argc != 3) {
```

```
    fprintf(stderr, "Usage: %s x y\n",
    argv[0]);
    return 1;
  }
  x = atof(argv[1]);
  y = atof(argv[2]);
  printf("min(%8.3f, %8.3f) = %8.3f\n", x,
   y, min(x, y));

  return 0;
}
```

```
double min(double x, double y)
{
  return (x > y) ? x : y;
}
```

编程

编写一个函数，打印一个字符矩阵，需要三个参数，即一个字符和两个整数，其中

- 字符参数是需要输出的字符
- 第一个整数为矩阵的行数
- 第二个整数为矩阵的列数

```
// print_char_mat.c
#include <stdio.h>
#include <stdlib.h>

void print_char_mat(int row, int col, char
 c);

int main(int argc, char * argv[])
{
  int row, col;
  char c;
```

```
if(argc != 4) {
   fprintf(stderr, "Usage: %s row col
   char", argv[0]);
   return 1;
}

row = atoi(argv[1]);
col = atoi(argv[2]);
c = argv[3][0];

print_char_mat(row, col, c);
```

```
  return 0;
}

void print_char_mat(int row, int col, char
 c)
{
  int i, j;
  for(i = 0; i < row; ++i) {
    for(j = 0; j < col; ++j)
      putchar(c);
    putchar('\n');
```

```
    }
}
```

```
$ gcc print_char_mat.c -o print_char_mat
$ ./print_char_mat 4 5 d
ddddd
ddddd
ddddd
ddddd
```

编程

编写一个函数，判断某一年是否为闰年。闰年的判断条件为：

1. 能够被 *4* 整除却不能被 *100* 整除的数。
2. 能够被 *400* 整除的数。

```c
// is_leap.c
#include<stdio.h>
#include<stdlib.h>
#include<stdbool.h>

bool is_leap(int year);

int main(int argc, char * argv[])
{
  int year;
  if(argc != 2) {
```

```
    fprintf(stderr, "Usage: %s year\n",
    argv[0]);
    return 1;
}

year = atoi(argv[1]);
if(is_leap(year))
    printf("%d is a leap year.\n", year);
else
    printf("%d is NOT a leap year.\n",
    year);
```

```
  return 0;
}

bool is_leap(int year)
{
  if(!year%4 && year%100 || !year%400)
    return true;
  return false;
}
```

```
$ gcc is_leap.c -o is_leap
$ ./is_leap 2018
2018 is NOT a leap year.
```

编程

已知三角形的三个顶点为 $(x_1, y_1), (x_2, y_2), (x_3, y_3)$，编写一个函数求三角形的面积，其中面积公式为

$$S = \frac{1}{2} \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{vmatrix}$$

```c
// is_leap.c
#include<stdio.h>
#include<stdlib.h>
#include<stdbool.h>

bool is_leap(int year);

int main(int argc, char * argv[])
{
  int year;
  if(argc != 2) {
```

```
    fprintf(stderr, "Usage: %s year\n",
    argv[0]);
    return 1;
}

year = atoi(argv[1]);
if(is_leap(year))
    printf("%d is a leap year.\n", year);
else
    printf("%d is NOT a leap year.\n",
    year);
```

```
  return 0;
}

bool is_leap(int year)
{
  if(!year%4 && year%100 || !year%400)
    return true;
  return false;
}
```

```
$ gcc find_triangle_area.c -o
find_triangle_area
$ ./find_triangle_area 0 0 1 0 0 1
Area =     0.500
```

编程

编写一个递归函数*power()*，用于计算一个 *double* 数的某正整数次幂。

编程

编写一个递归函数*power()*，用于计算一个 *double* 数的某正整数次幂。

$$a^n = \begin{cases} 1, & n = 0, \\ \left(a^{\lfloor \frac{n}{2} \rfloor}\right)^2 \times a, & n \text{ odd}, \\ \left(a^{\lfloor \frac{n}{2} \rfloor}\right)^2, & n \text{ even} \end{cases}$$

```c
#include<stdio.h>
#include<stdlib.h>

double power(double a, int n)
{
  if(0 == n)
    return 1.0;
  if(n % 2) // odd
    return power(a, n/2)*power(a, n/2)*a;
  else
    return power(a, n/2)*power(a, n/2);
```

```
}


int main(int argc, char * argv[])
{
  int n;
  double a;

  if (argc != 3) {
    printf("usage: %s base exponetial\n",
    argv[0]);
    return 1;
  }
```

```
    a = atof(argv[1]);
    n = atoi(argv[2]);
    printf("%.2f ^ %2d = %.2f\n", a, n,
    power(a, n));

    return 0;
}
```

```
$ gcc power.c -o power
$ ./power 2 3
2.00 ^ 3 = 8.00
```

编程

回顾十进制转二进制的函数*to_binary()*，然后思考一下如何对其进行推广，将某个十进制数转换为任意进制，即构造一个新的函数*dec2base()*。如*dec2base(129, 16)*的输出为*7F*，即*129*的十六进制数。

```
// dec2base.c
#include <stdio.h>
#include <stdlib.h>

char *BaseString = "0123456789ABCDEF";

void dec2base(unsigned long dec, int base)
;

int main(int argc, char * argv[])
{
```

```
unsigned long dec;
int base;

if(argc != 3) {
   fprintf(stderr, "Usage: %s dec base\n"
   , argv[0]);
   return 1;
}

dec = strtoul(argv[1], NULL, 0);
base = atoi(argv[2]);
printf("(10) %lu = (%d) ", dec, base);
```

```
    dec2base(dec, base);
    putchar('\n');

    return 0;
}

void dec2base(unsigned long dec, int base)
{
    int r;
    r = dec % base;
    if (dec  >= base)
        dec2base(dec / base, base);
```

```
    putchar(BaseString[r]);
    return;
}
```

```
$ gcc dec2base.c -o dec2base
$ ./dec2base 32 16
(10) 32 = (16) 20
```

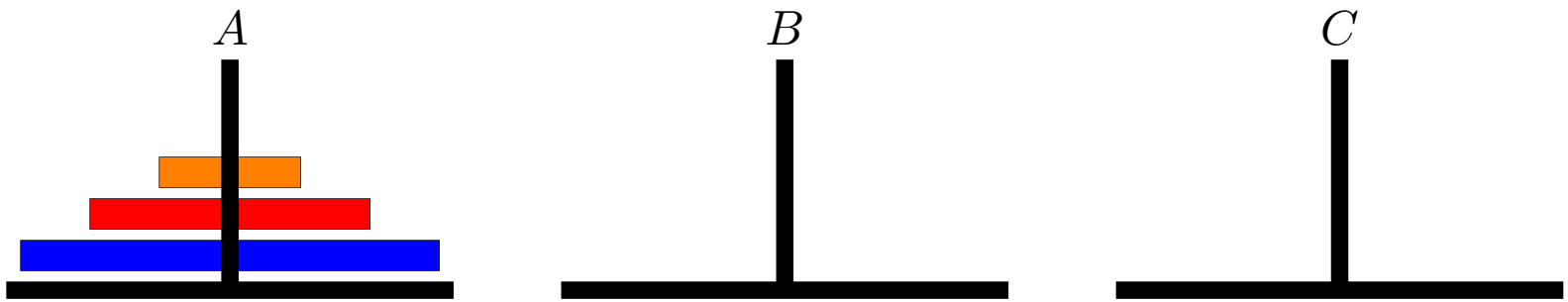
编程

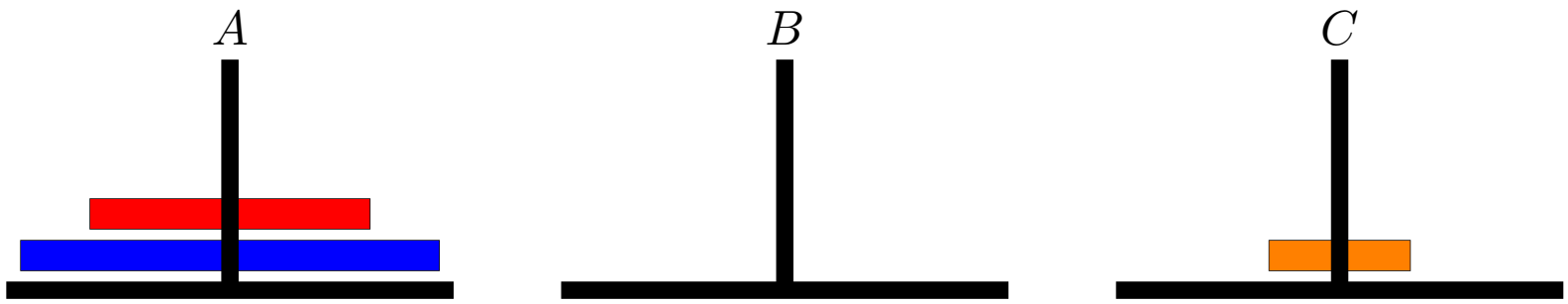
传说婆罗门庙里有一个塔台，台上有 *3* 根标号为 *A*、*B*、*C* 的用钻石做成的柱子，在 *A* 柱上放着 *64* 个金盘，每一个都比下面的略小一点。把 *A* 柱上的金盘全部移到 *C* 柱上的那一天就是世界末日。移动的条件是：

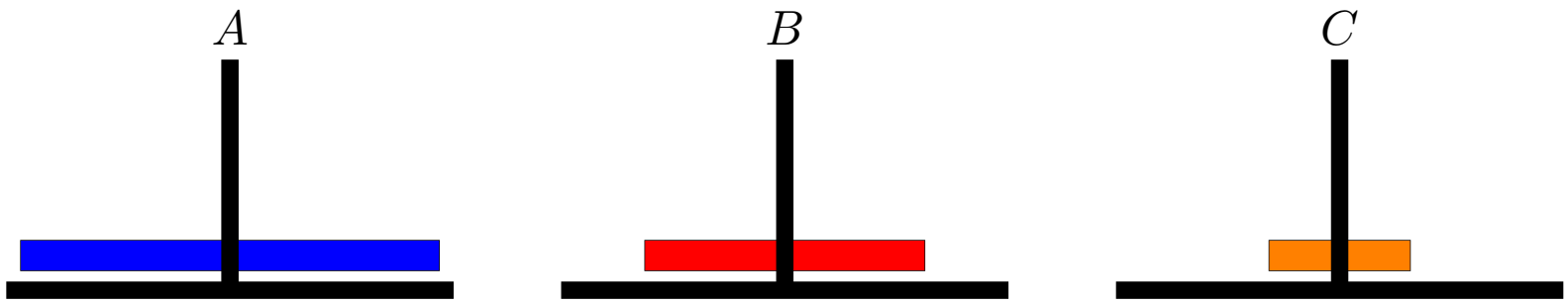
- 一次只能移动一个金盘；
- 移动过程中大金盘不能放在小金盘上面。

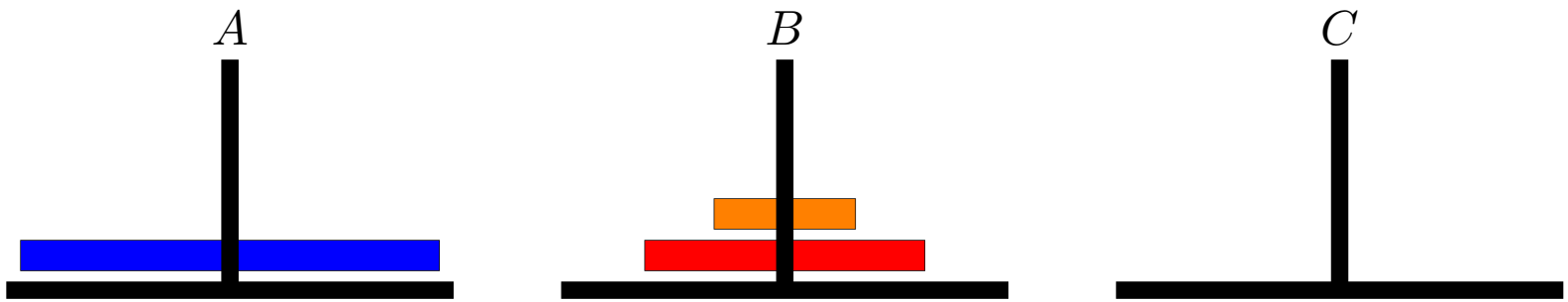
庙里的僧人一直在移个不停，移动的最少总次数是 $2^{64}-1$ 次，如果每秒移动一次的话，需要 *500* 亿年。

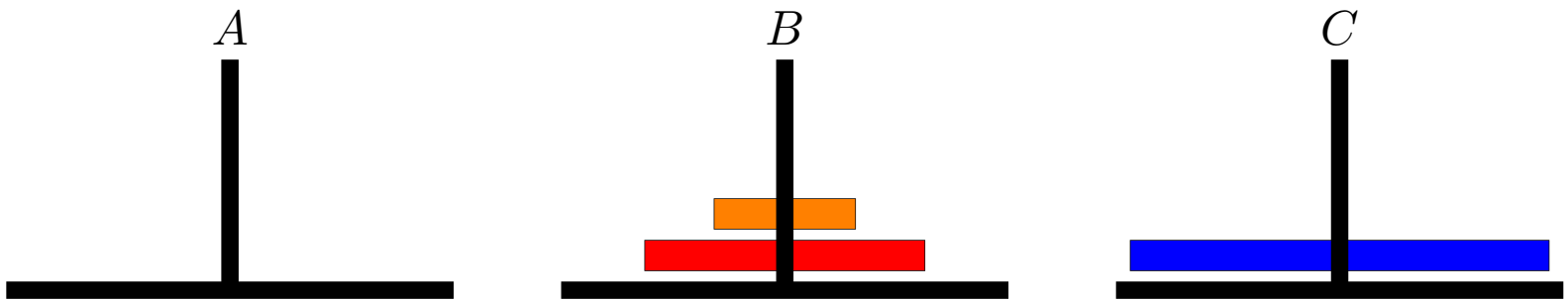
- 若 $n=1$，则将这一个盘子直接从 A 柱移动到 C 柱上，最少移动 $2^1-1=1$ 次。
- 若 $n>1$，则执行以下 3 步，最少移动 2^n-1 次：
  1. 借助 C 柱，将 A 柱上的 $n-1$ 个盘子移到 B 柱；
  2. 将 A 柱上最后一个盘子直接移到柱；
  3. 借助 A 柱，将 B 柱上的 $n-1$ 个盘子移到 C 柱。

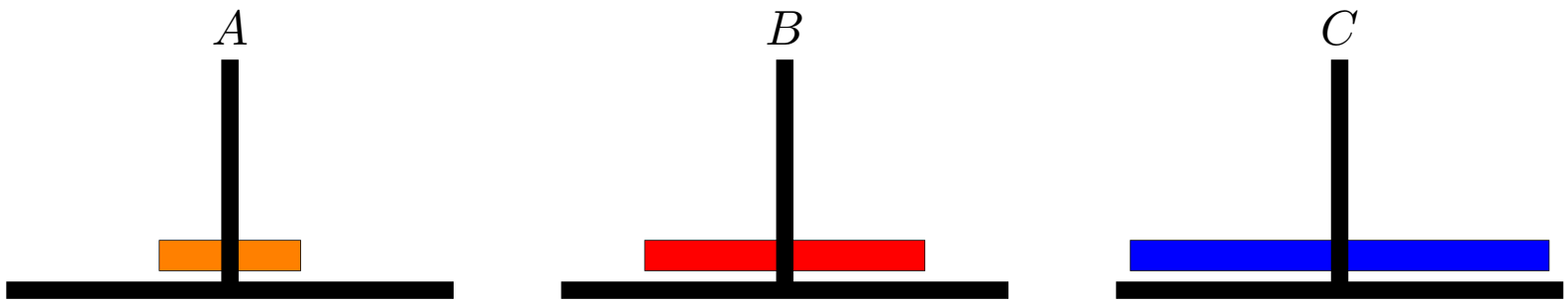

A
B
C

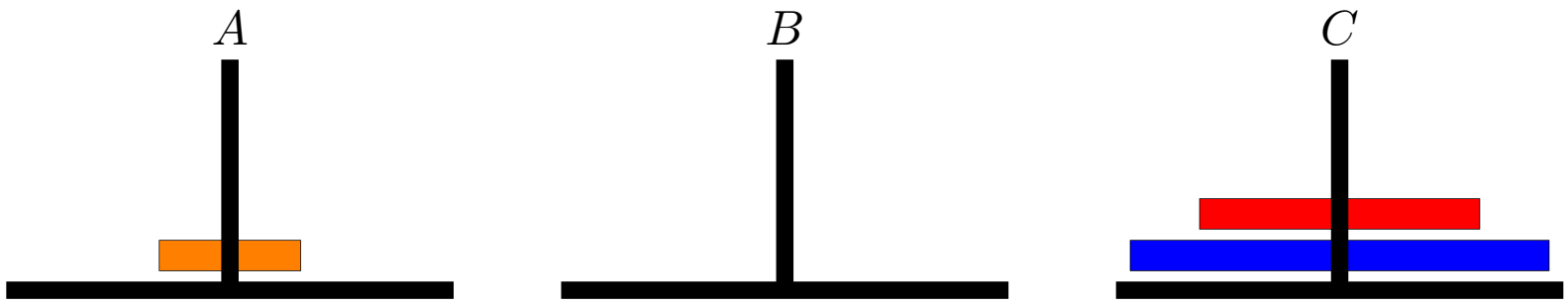

A
B
C

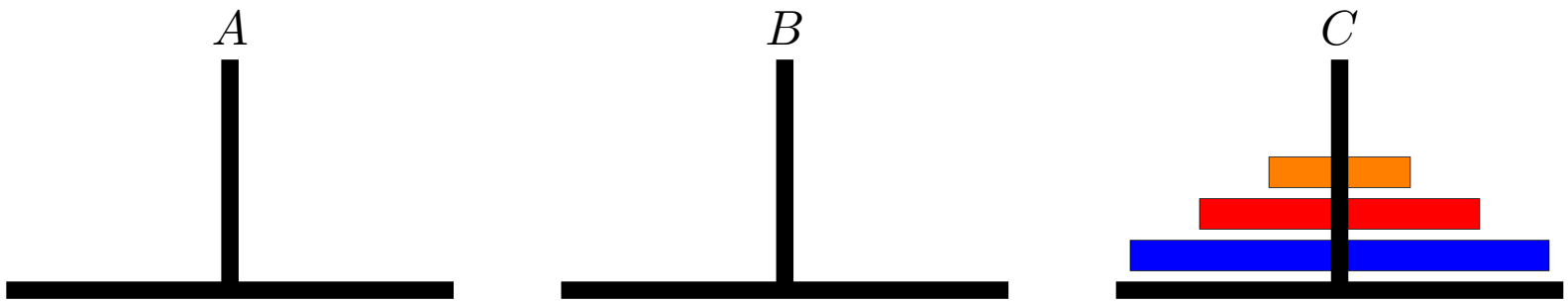

A
B
C


A
B
C


A
B
C


A
B
C


A
B
C


A
B
C

```c
// hanoi.c
#include <stdio.h>
#include <stdlib.h>

typedef struct {
  int arr[15];
  int nr;
} Stick;

Stick data[3];
```

```
void hanoi_init(int n)
{
  int i;
  for(i = 0; i < n; i++)
    data[0].arr[i] = n - i;
  data[0].nr = n;
  data[1].nr = 0;
  data[2].nr = 0;
}

void hanoi_display(void)
{
```

```
  int i, j, k;
  for(i = 0 ;i < 3; i++) {
    for(j = 0; j < data[i].nr; j++)
      printf("%2.d", data[i].arr[j]);

    for(k = data[i].nr; k >= data[i].nr &&
     k < 15; k++)
      printf(" -");
    printf("\n");
  }
}
```

```
void move(int src, int dest)
{
  int tmp;
  tmp = data[src].arr[--data[src].nr];
  data[dest].arr[data[dest].nr++] = tmp;
  getchar();
  hanoi_display();
}


void hanoi(int src, int dest, int tmp, int
 n)
```

```
{
  if(n == 1) {
    move(src, dest);
    return;
  }
  hanoi(src, tmp, dest, n-1);
  move(src, dest);
  hanoi(tmp, dest, src, n-1);
}

int main(int argc, char **argv)
{
```

```
if(argc != 2) {
  printf("usage: %s <1-15>\n", *argv);
  return 0;
}
else {
  int m = atoi(argv[1]);
  if(m < 1 || m > 15) {
    printf("pls input a num < 0 - 15 >
    not %d\n", m);
    return 0;
  }
  hanoi_init(m);
```

```
    hanoi_display();
    hanoi(0, 2, 1, m);
  }
  return 0;
}
```