### Bài toán tháp Hà nội

Có 3 chiếc cọc và một bộ n chiếc đĩa. Các đĩa này có kích thước khác nhau và mỗi đĩa đều có 1 lỗ ở giữa để có thể xuyên chúng vào các cọc. Ban đầu, tất cả các đĩa đều nằm trên 1 cọc, trong đó, đĩa nhỏ hơn bao giờ cùng nằm trên đĩa lớn hơn.

C

ọ

c A

C

ọ

c B

C

ọ

c C

#### **Hình 2.2** Bài toán tháp Hà nội

Yêu cầu của bài toán là chuyển bộ n đĩa từ cọc ban đầu A sang cọc đích C (có thể sử dụng cọc trung gian B), với các điều kiện:

* Mỗi lần chuyển 1 đĩa.
* Trong mọi trường hợp, đĩa có kích thước nhỏ hơn bao giờ cũng phải nằm trên đĩa có kích thước lớn hơn.

#include<stdio.h>

void chuyen(int n, char a, char c);

void thaphanoi(int n, char a, char c, char b);

void chuyen(int n, char a, char c){

printf("Chuyen dia thu %d tu coc %c sang coc %c \n", n,

a, c);

return;

}

void thaphanoi(int n, char a, char c, char b){

if (n==1) chuyen(1, a, c);

else{

thaphanoi(n-1, a, b, c);

chuyen(n, a, c);

thaphanoi(n-1, b, c,a);

}

return;

}

int main(){

int sodia;

printf("Cho biet so dia can chuyen: ");

scanf("%d",&sodia);

printf("\nCac buoc chuyen nhu sau:\n\n");

thaphanoi(sodia, 'a', 'c', 'b');

return 0;

}

Lab 2

1. Viết lại specification, vẽ lại diagram, viết lại code từ C -> Java/.Net => vẽ diagram của code (Docx)

2. Viết test cases

Input = / Expected Result = / Actual Result = dùng biến, ko hard code

2 phương pháp:

+ Blackbox: specification-derived test, equivalence partitioning, boundary value analysis

(input/output)

+ Whitebox: statement coverage, decision coverage, path coverage

(input/output/structure)

3. Comment

- Phương pháp gì/ kỹ thuật gì //Blackbox //Specification

- index test case

Vd: TC1 - Statement 1