Trường Đại học Khoa học Tự nhiên

Khoa Toán – Cơ – Tin học

Học kỳ II – 2021 - 2022

Thi cuối kỳ: môn **Lập trình Java**

Thời gian làm bài: 120 phút

Đề 1

Bài 1 (1 điểm). Để tính căn bậc hai của một số nguyên dương x người ta có thể sử dụng thuật toán Babylonian như sau:

Thuật toán Babylonian: Để tính căn bậc hai của một số nguyên dương x, ta có thể sử dụng thuật toán sau:

1. Đặt $y = 1.0$
2. Thay y với trung bình cộng của y và x/y.
3. Lặp lại bước 2 đến khi khoảng cách giữa y và x/y đủ nhỏ.
4. Trả về y.

Viết chương trình nhập vào một số nguyên x từ bàn phím, sử dụng thuật toán Babylonian để tính xấp xỉ căn bậc hai của x với độ chính xác 10^{-6}.

Ví dụ:

```
Input x (x > 0): 7

Babylonian Algorithm: 2.6457513110645907
```

Bài 2 (2 điểm). Để tính gần đúng tích phân xác định, người ta có thể sử dụng công thức hình thang được mô tả như sau:

Công thức hình thang (trapezoidal rule): Tích phân xác định có thể được tính xấp xỉ sử dụng công thức hình thang như sau:

$$\int_a^b f(x)dx \approx h\left[\frac{f(x_0)+f(x_1)}{2}+\frac{f(x_1)+f(x_2)}{2}+\cdots+\frac{f(x_{n-1})+f(x_n)}{2}\right]$$

$$= h\left[\frac{f(x_0)+f(x_n)}{2}+f(x_1)+\cdots+f(x_{n-1})\right] = l_n,$$

trong đó $h = \frac{b-a}{n}, x_i = a + ih.$

Dùng công thức hình thang, tính gần đúng tích phân xác định sau với độ chính xác 10^{-6}:

$$\int_a^b \sqrt{x^2+1}dx,$$

Độ chính xác ở đây được xác định như sau, chọn $n_0 = 2$, sau đó tính l_n với $n = n_0, 2n_0, 4n_0, \ldots$ Việc tính toán dừng lại khi $|l_{2n} - l_n| < \epsilon = 10^{-6}$.

Ví dụ:

```
Input a and b: 0 1

Trapezoidal Rule: 1.1477937994795209
```

Bài 3 (3 điểm). File countries1.csv lưu dữ liệu về tên nước, dân số, diện tích, gdp, khu vực của một số nước trên thế giới. Chương trình quản lý thông tin các nước từ dữ liệu trong file countries1.csv, được thiết kế như sau:

- Lớp **CountryData** chứa dữ liệu một nước, tương ứng với một dòng dữ liệu được đọc vào từ file countries1.csv.
- Lớp **CountryArrayManager** là lớp để quản lý dữ liệu các nước (được lưu trong **CountryData**). Lớp **CountryArrayManager** sử dụng cấu trúc dữ liệu kiểu mảng để quản lý dữ liệu.
- Lớp **App** là lớp client để thực thi chương trình.

Viết code để hoàn thiện chương trình theo thiết kế đã cho trong các file source code đi kèm.

Bài 4 (3 điểm). File countries2.csv lưu dữ liệu về tên nước, dân số, diện tích, gdp, khu vực của một số nước trên thế giới. Chương trình quản lý thông tin các nước từ dữ liệu trong file countries2.csv, được thiết kế như sau:

- Lớp **CountryData** chứa dữ liệu một nước, tương ứng với một dòng dữ liệu được đọc vào từ file countries2.csv.
- Lớp **AbstractCountry** chứa những đặc điểm chung của các nước.
- Lớp **Country** là một lớp chứa đầy đủ thông tin chi tiết về một nước, được thừa kế từ lớp **AbstractCountry**.
- Lớp **CountryListManager** là lớp để quản lý dữ liệu các nước (được lưu trong **Country**). Lớp **CountryListManager** sử dụng cấu trúc dữ liệu kiểu **List** để quản lý dữ liệu.
- Lớp **App** là lớp client để thực thi chương trình.

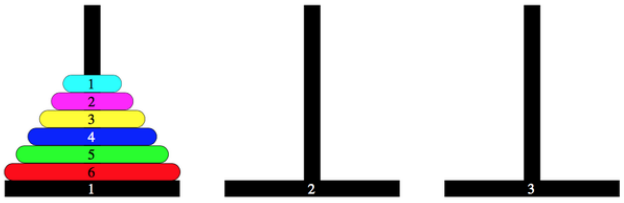
Viết code để hoàn thiện chương trình theo thiết kế đã cho trong các file source code đi kèm.

Bài 5 (1 điểm). Bài toán tháp Hà Nội là một trò chơi nổi tiếng với việc sử dụng thuật toán đệ quy để giải. Luật chơi được mô tả như sau:

Trò chơi này gồm một bộ N đĩa kích thước khác nhau, có lỗ ở giữa, nằm xuyên trên ba cái cột. Bài toán đố bắt đầu bằng cách sắp xếp các đĩa theo trật tự kích thước vào một cột, sao cho đĩa nhỏ nhất nằm ở trên cùng, tức là tạo thành một hình nón.

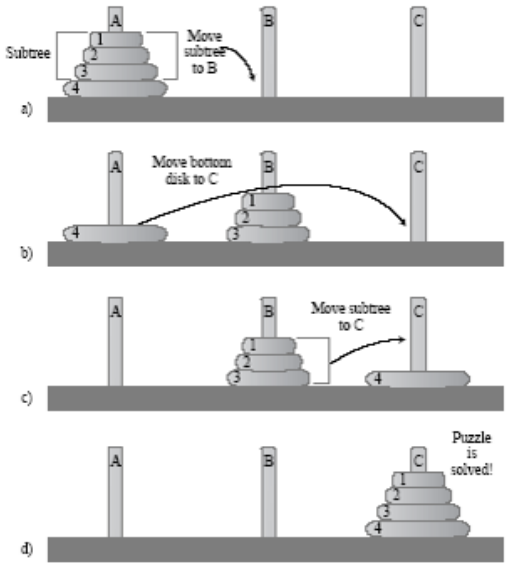
Yêu cầu của trò chơi là di chuyển toàn bộ số đĩa sang một cột khác, tuân theo các quy tắc sau:

- Một lần chỉ có 3 cột để di chuyển, gọi là A, B, C.
- Chỉ di chuyển một đĩa trên cùng (không được di chuyển đĩa nằm giữa hay nằm cuối).
- Một đĩa chỉ có thể đặt lên một đĩa lớn hơn (không nhất thiết hai đĩa này phải có kích thước liền kề, tức là đĩa nhỏ nhất vẫn có thể đặt trên đĩa lớn nhất).



Ý tưởng để giải bài toán sử dụng thuật toán đệ quy như sau:

- Nếu đã biết cách chuyển N - 1 đĩa từ cột A sang cột B, ta chỉ cần chuyển đĩa thứ N (đĩa cuối cùng) từ cột A sang cột C, rồi chuyển N - 1 đĩa từ cột B sang cột C.
- Giải thuật không còn đệ quy khi chỉ có 1 đĩa, vì ta chuyển trực tiếp từ cột A sang cột C luôn mà không cần thông qua cột B.



Thuật toán đệ quy để giải bài toán như sau:

Algorithm 1: Recursive algorithm for solving Towers of Hanoi

```
1 function recursiveHanoi(n, s, a, d)
2     if n == 1 then
3         print(s + " to " + d);
4         return;
5     end
6     recursiveHanoi(n − 1, s, d, a);
7     print(s + " to " + d);
8     recursiveHanoi(n − 1, a, s, d);
9 end
```

Viết chương trình giải bài toán sử dụng thuật toán đệ quy trên.

Ví dụ:

```
n = 3
D1 A to C
D2 A to B
D1 C to B
D3 A to C
D1 B to A
D2 B to C
D1 A to C
```

Một số lưu ý khi làm bài:

- Làm bài bằng cách hoàn thiện các phương thức đã được thiết kế trước.
- Không thay đổi nguyên mẫu (prototype) các phương thức đã được thiết kế trước. Viết thêm các phương thức còn thiếu. Được phép viết thêm các hàm phụ.
- Để "pass" các "test case", kết quả in ra phải trùng khớp với output trong các test case. Lỗi hay gặp phải là in ra thừa ký tự trắng hoặc xuống dòng.
- Các test case chỉ được sử dụng để tham khảo, khi chấm bài căn cứ vào code để chấm. Code xấu, vi phạm nhiều coding convention bị trừ 20% số điểm.