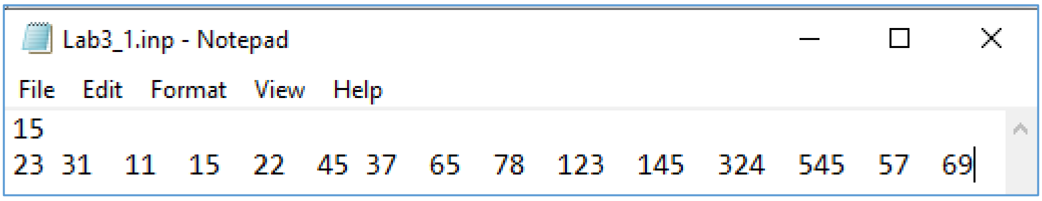
BÀI THỰC HÀNH 3

Nội dung:

- Hàm đệ qui
- Kỹ thuật quay lui

Bài 3.1 Dùng NotePad tạo file lưu trữ số phần tử của mảng gồm n số nguyên có tên "*Lab3_1.inp*" như sau:



Viết các hàm thực hiện các yêu cầu sau:

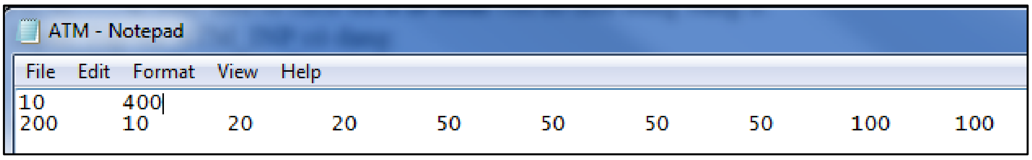
a) Đọc nội dung trong file "Lab3_1.inp" ra mảng a;
b) Xuất mảng a ra màn hình;
c) Tìm và xuất ra màn hình số lớn nhất trong mảng a (dùng vòng lặp, đệ qui);
d) Tính tổng các phần tử của mảng a (dùng đệ qui đầu, đệ qui đuôi)
e) Hàm main() gọi các hàm trên thực hiện để kiểm tra kết quả.

Bài 3.2 Viết hàm dùng vòng lặp, đệ qui thực hiện các yêu cầu:

a) Tính $S(n) = a^n$, với ($1 < n < 30$)
b) Tìm số fibonacii thứ n ($3 < n < 20$);
c) Bài toán tháp Hà Nội;
d) Hàm main() gọi các hàm trên thực hiện để kiểm tra kết quả.

Bài 3.3. Bài toán rút tiền ATM (tìm một nghiệm). Một máy ATM hiện có n ($n \le 20$) tờ tiền với mệnh giá $t_1, t_2, \ldots, t_n$. Hãy đưa ra các cách trả tiền với số tiền đúng bằng S *(dùng kỹ thuật quay lui)*

- Đầu vào file văn bản (*ATM.inp*) chứa 2 dòng
    - Dòng đầu chứa giá trị n, s ($n<30$)
    - Dòng sau chứa mệnh giá trị của các tờ tiền $t_1, t_2, .., t_n$



- Đầu ra: file văn bản (*ATM_out.out*) nếu có thể trả đúng s thì đưa ra cách trả, không có cách trả thì ghi -1.

Bài 3.4. Phân tích số

- Hãy tìm tất cả các cách phân tích số n ($0< n < 20$) thành tổng của các số nguyên dương, các cách phân tích là hoán vị của nhau chỉ tính là 1 cách.
  Kết quả phân tích ghi ra file văn bản (*Lab4_3.out*), mỗi cách viết trên 1 dòng; nếu không có cách phân tích thì ghi -1.

Ví dụ: nhập vào n = 5

```
File   Edit   Format   View
1 + 1 + 1 + 1 + 1
1 + 1 + 1 + 2
1 + 1 + 3
1 + 2 + 2
2 + 3
5
```

Ý tưởng: Ta sẽ lưu nghiệm trong mảng x, ngoài ra có một mảng s để lưu tổng các phần tử đến bước đang xét. Mảng s xây dựng như sau: s_i sẽ là tổng các phần tử trong mảng x từ x_1 đến x_i: $s_i := x_1 + x_2 + ... + x_i$.

Khi liệt kê các dãy x có tổng các phần tử đúng bằng n, để tránh sự trùng lặp ta đưa thêm ràng buộc $x_{i-1} \le x_i$. Khi $s_i = n$ tức là ($x_i = n - s_{i-1}$) thì in kết quả.

Vậy thủ tục Try(i) thử các giá trị cho x_i có thể mô tả như sau: (để tổng quát cho i = 1, ta đặt x_0=1 và s_0= 0).

- Xét các giá trị của x_i từ x_{i-1} đến ($n - s_{i-1}$), cập nhật $s_i := s_{i-1} + x_i$ và gọi đệ quy tìm tiếp.

Bài 3.5* Hãy in ra số lượng cách để phân tích số n thành tổng các số nguyên dương. Để tìm các cách phân tích số n, ta có thể nghĩ đến việc chọn một số x trong khoảng từ 1 đến n, sau đó tìm cách phân tích phần còn lại (n - x). Như vậy, vấn đề được chia thành các bài toán con nhỏ hơn.

Ý tưởng:

- Gọi hàm đệ quy countPartitions(n, max) để tính số cách phân tích số n thành tổng các số nguyên dương không lớn hơn max.
- Từ đó, số cách phân tích số n sẽ được tính bằng cách chia nhỏ thành các trường hợp:
    - Trường hợp không chọn số lớn hơn max: phân tích số còn lại là n - max với các số nhỏ hơn hoặc bằng max.
    - Giảm dần max để thử với các số nhỏ hơn.

Công thức đệ quy:

- Nếu n == 0: Khi đã phân tích xong (không còn số nào cần phân tích), ta có một cách hợp lệ, trả về 1.
- Nếu n > 0: Ta có thể chọn một số x từ 1 đến max và sau đó tìm cách phân tích phần còn lại n - x. Ta thử với các số x từ 1 đến max.
- Nếu max == 0: Trả về 0 vì không còn cách nào để phân tích.

Điều kiện cơ sở:

- Nếu n == 0: Ta trả về 1 vì chỉ có một cách là không chọn thêm số nào nữa.
- Nếu max == 0 hoặc n < 0: Trả về 0 vì không thể có cách phân tích hợp lệ.

Thuật toán:

1. Bắt đầu với tổng n cần phân tích và số lớn nhất max là n.
2. Gọi hàm đệ quy để phân tích số n với các số nhỏ hơn hoặc bằng max.

Bài 3.6* Bài toán tìm đường trong mê cung

Cho một mê cung được biểu diễn bởi một ma trận nhị phân n x n, trong đó:

- 1 biểu thị ô có thể đi qua.
- 0 biểu thị ô không thể đi qua.

Nhiệm vụ là tìm một đường đi từ ô bắt đầu (0, 0) đến ô kết thúc (n-1, n-1) nếu có; tại mỗi ô chỉ có thể di chuyển theo bốn hướng: lên, xuống, trái, phải. Nếu không có đường đi, in ra "Không có đường đi".

Ví dụ: Cho mê cung là ma trận sau:

1 0 0 0

1 1 0 1

0 1 0 0

1 1 1 1

Một đường đi hợp lệ sẽ là:

(0,0) -> (1,0) -> (1,1) -> (2,1) -> (3,1) -> (3,2) -> (3,3)

Ý tưởng:

Sử dụng phương pháp **quay lui (backtracking)** để thử từng bước đi, kiểm tra xem bước đi có hợp lệ hay không. Nếu gặp ngõ cụt, quay lui lại bước trước đó và thử hướng đi khác.

Các bước:

1. Bắt đầu từ ô (0, 0), di chuyển theo bốn hướng (lên, xuống, trái, phải).
2. Kiểm tra xem ô có thể di chuyển được không (ô có giá trị 1 và nằm trong giới hạn của mê cung).
3. Nếu ô hiện tại là điểm kết thúc (n-1, n-1), nghĩa là đã tìm được đường đi.
4. Nếu không thể đi tiếp ở hướng hiện tại, quay lui và thử hướng khác.
5. Nếu tất cả các hướng đều không dẫn tới đích, kết luận rằng không có đường đi.

----- hết -----