Họ và tên: NGUYỄN THÀNH LONG

MSSV: 20120324

Mã lớp: 20CTT2

FILE BÁO CÁO XỬ LÝ BIGINT

Giới thiệu các biến quan trọng trong chương trình:

NUM: là struct dùng để lưu mọi thông tin của một big int:

* Sign : lưu dấu ( dương ứng với 0, âm ứng vs 1)
* lengthNum: lưu độ dài kí tự chữ số của big int(để thuận tiện cho so sánh)
* ListNum: danh sách liên kết đơn mỗi node chứa k kí tự chữ số( k nhập vào từ tham số dòng lệnh) lưu giá trị từ cuối về đầu. VD big int = 1234567, k =3 ->thứ tự lưu từ pHead-> pTail là: 567->234->1

ListInt: là danh sách liên kết đơn mà data ở đây là NUM, danh sách liên kết này dung để lưu các giá trị big int thành một danh sách liên kết thuận tiện cho việc sort.

1. Chức năng cộng, trừ và so sánh

* **Chức năng cộng**: add(NUM a, NUM b, int k)

khai báo NUM x là kết quả của phép cộng NUM a và NUM b

Trường hợp nếu a và b cùng dấu

+ Nếu a âm(a.sign = 1) và b âm(b.sign = 1) thì lưu dấu kết quả là âm(x.sign = 1) và tính a + b. (vì b nhập vào luôn là số dương nên trường hợp này xảy ra là hệ quả của phép trừ ( -a) – b chuyển thành (–a) + (-b) )

+ Nếu a và b cùng dương (a.sign = 0 và b.sign = 0) thì lưu dấu kết quả là dương(x.sign = 0) và tính a + b

Trường hợp a và b khác dấu:

+ Chỉ xảy ra trường hợp a âm và b dương -> trả về kết quả b – a(đưa về hàm trừ).

Trình tự xử lí: (vì dấu của phép tính đã được xử lí rồi nên code bên dưới chỉ là thực hiện phép tính)

+ so sánh chiều dài kí tự của a và b, đổi số có chiều dài lớn hơn về a và số còn lại về b.

+ SumData dung để lưu giá trị tổng node tương ứng của a và b.

+biến du dung để lưu số dư nếu SumData > chiều dài node được lưu- cụ thể là lớn hơn hoặc bằng 10^k.

+ i dung để duyệt ListNum của a từ pHead tới Ptail

+ j dung để duyệt ListNum của b tương ứng a, và b sẽ luôn kết thúc trước hoặc cùng lúc với a.

+ cộng và xử lí số dư tại các node.

+duyệt để cập nhật x.lengthNum.

+trả về x.-> kết thúc chương trình.

* **Chức năng trừ**:subtract(NUM a, NUM b, int k)

Chỉ có 2 trường hợp: -a – b (=> đưa về cộng 2 số âm) hoặc a – b => hàm trừ chỉ xử lí trường hợp này.

Trình tự xử lí:

+ Nếu a bằng b(compare(a,b,k) = 2) thì kết quả là 0.

+ Nếu a < b(compare(a,b,k) = 0) thì chuyển thành b – a và gán dấu âm cho kết quả.

+ Nếu a > b(compare(a,b,k) = 1) thì tính a - b.

+subData dung để lưu giá trị hiệu tương ứng của các node a và b

+ no dung để ghi nợ nếu tại node tương ứng a < b => a phải mượn 1 ở đầu để trừ b.

+ i và j giống hàm cộng

+ trừ và xử lí nợ tại các node.

+duyệt để cập nhật x.lengthNum.

+trả về x.-> kết thúc chương trình.

* **Chức năng so sánh:** compare(NUM a, NUM b, int k)

Nếu a nhỏ hơn b trả về 0

Nếu a lớn hơn b trả về 1

Nếu a bằng b trả về 2

Vì b luôn dương nên có các trường hợp sau:

+ a âm(a.sign == 0) trả về 0.

+ a và b cùng dương

Nếu a dài hơn b => trả về 1

Nếu a ngắn hơn trả về 0

Nếu a dài bằng b => so sánh từng node tương ứng bằng cách reverseNum a và b.(có chú thích bên dưới).

Nếu có 1 node tương ứng của a lớn hơn b => trả về 1

Nếu có 1 node tương ứng của a nhỏ hơn b => trả về 0

+ Nếu tất cả các trường hợp trên đều không xảy ra thì trả về 2.

* **Các hàm phụ quan trọng:**

NumToString(NUM x, int k): chuyển Num x về lại string để ghi file.

reverseNum(NUM x): đảo ngược ListNum , đảo node từ Tail lên Head theo thứ tự. VD 567 234 1 -> 1 234 567.

1. Chức năng sort:

Sử dụng ListInt để lưu trữ danh sách liên kết với các data là NUM.

Dung quicksort và MergeSort với ý tưởng như thầy đã dạy.