ĐỀ KIỂM TRA GIỮA KỲ **MÔN: CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT** Thời gian: **60 phút** (**KHÔNG** sử dụng tài liệu)	23

Câu 1: (*4 điểm*) Cho ma trận A (nXm) chỉ chứa các giá trị 0 và -1. Giả sử có định nghĩa về "thành phần liên thông" như sau: một nhóm các ô liên tục (chỉ tính 4 phía trên, dưới, trái, phải) cùng chứa giá trị -1 được xem là 1 "thành phần liên thông". Viết hàm đếm số "thành phần liên thông" có trong ma trận.

Minh họa:

0	-1	-1	0	0	0	-1
0	-1	-1	0	0	0	0
0	-1	0	0	0	0	0
0	-1	-1	0	0	0	-1
0	0	0	0	0	0	-1
0	-1	0	-1	-1	-1	-1
0	-1	0	0	0	0	-1
Có 4 thành phần liên thông						

0	0	-1	0
0	-1	-1	-1
0	0	-1	0
0	0	0	0
-1	0	0	0
0	-1	0	0
0	0	0	0
Có 3 TPLT			

Câu 2: (*3 điểm*) Cho khai báo cấu trúc như sau:

```
typedef struct Node
{
	int Key;
	Node* pNext;
}NODE;
typedef struct List
{
	Node* pHead;
	Node* pTail;
}LIST;
```

Giả sử danh sách liên kết đã được tạo trước đó. Viết hàm xóa tất cả các node có chứa *Key* là số Armstrong.

📖*Số Armstrong*

- Là số có đặc điểm sau: số đó gồm n ký số, tổng các lũy thừa bậc n của các ký số bằng chính số đó.
- *Ví dụ* *8 (số có 1 ký số), và $8^1 = 8$.*
  153 (số có 3 ký số), và $1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153$.
  1634 (số có 4 ký số), và $1^4 + 6^4 + 3^4 + 4^4 = 1+1296+81+256=1634$.

Câu 3: (*3 điểm*) Từ cây nhị phân tìm kiếm cân bằng ban đầu là trống, người ta đưa dãy số sau vào cây: `17, 4, 9, 5, 6, 22, 91, 18, 32, 45, 87, 35, 16, 4`.

Lần lượt vẽ lại cây qua mỗi lần cần thực hiện cân bằng khi đưa các số trên vào cây.

- HẾT –

ĐỀ KIỂM TRA GIỮA KỲ MÔN: CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT Thời gian: **60 phút** (**KHÔNG** sử dụng tài liệu)	14

Câu 1: (*3 điểm*) Cho mảng a gồm n phần tử và mảng b gồm m phần tử, hai mảng đã được sắp xếp theo thứ tự không giảm.

Viết hàm thực hiện yêu cầu sau: Với mỗi phần tử trong mảng b, đếm và in ra số lượng phần tử trong mảng a có giá trị nhỏ hơn phần tử (đang xét) trong mảng b.

Hàm được đề xuất có dạng như sau:

```
void Dem (int a[], int n, int b[], int m)
{   // SV bổ sung lệnh vào đây để thực hiện yêu cầu.
}
```

Biết rằng:

- $1 <= n, m <= 10^6$
- $-10^9 <= a_i, b_i <= 10^9$

- ***Ví dụ***: với
```
a = {1, 6, 9, 13, 18, 18};        n=6
b = {2, 3, 8, 13, 15, 21, 25};    m=7
```
chương trình sẽ xuất ra:
```
1, 1, 2, 3, 4, 6, 6
```

YÊU CẦU: viết hàm thực hiện yêu cầu trên sao cho độ phức tạp đạt là O(n).

Câu 2: (*4 điểm*) Cho khai báo cấu trúc như sau:

```
struct TNODE
{
    int Key;
    struct TNODE *pLeft,
    struct TNODE *pRight;
} *TREE;
```

Giả sử cây nhị phân tìm kiếm đã được tạo trước đó. Viết hàm xóa tất cả các node có chứa *Key* đều là số thân thiện.

📖*Số thân thiện*:

- Là những số mà số đó và số đảo ngược của nó sẽ có ước chung lớn nhất là 1.
- *Ví* dụ với số 23, số đảo ngược của nó là 32. Hai số này có ước chung lớn nhất là 1. Khi đó cả 2 số 23 và 32 đều được gọi là số thân thiện.
- Các số thân thiện có giá trị nhỏ hơn 100 gồm: *1, 10, 13, 14, 16, 17, 19, 23, 25, 29, 31, 32, 34, 35, 37, 38, 41, 43, 47, 49, 52, 53, 56, 58, 59, 61, 65, 67, 71, 73, 74, 76, 79, 83, 85, 89, 91, 92, 94, 95, 97, 98.*

Câu 3: (*3 điểm*)

3.1 Đưa dãy số sau vào minHeap: `99, 88, 77, 66, 55, 44, 33, 22, 11`.

3.2 Vẽ lại Heap sau khi xóa xong số `55`.

- HẾT –

ĐỀ KIỂM TRA GIỮA KỲ **MÔN: CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT** Thời gian: **60 phút** (**KHÔNG** sử dụng tài liệu)	45

Câu 1: (*3 điểm*) Viết hàm nhận tham số là số nguyên dương n (n>0). Hàm thực hiện loại bỏ 1 số trong n sao cho giá trị các số còn lại trong n là nhỏ nhất.

Ví dụ: n = 21 ⇒ bỏ số 2 ⇒ n = 1
n = 132 ⇒ bỏ số 3 ⇒ n = 12
n = 104 ⇒ bỏ số 1 ⇒ n = 4
n = 23198 ⇒ bỏ số 3 ⇒ n = 2198

Gợi ý: lần lượt xét từng cặp số từ trái sang phải, khi số đi trước lớn hơn số đi sau thì loại bỏ số đi trước.

Câu 2: (*4 điểm*) Cho khai báo cấu trúc như sau:

```
struct TNODE
{
    int Key;
    struct TNODE *pLeft,
    struct TNODE *pRight;
} *TREE;
```

Giả sử cây nhị phân tìm kiếm đã được tạo trước đó. Viết hàm đếm tất cả các node lá có *Key* là số may mắn.

📖 ***Lucky Number***

- (Theo Wikipedia) *Lucky Number (số may mắn)* là số được định nghĩa theo quá trình sau: bắt đầu với số nguyên dương x và tính tổng bình phương y các chữ số của x, sau đó tiếp tục tính tổng bình phương các chữ số của y. Quá trình này lặp đi lặp lại cho đến khi thu được kết quả là 1 thì dừng (tổng bình phương các chữ số của số 1 chính là 1) hoặc quá trình sẽ kéo dài vô tận. Số mà quá trình tính này kết thúc bằng 1 gọi là số may mắn. Số có quá trình tính kéo dài vô tận là *số không may mắn* hay còn gọi là ***sad number*** (*số đen đủi*). Ví dụ: 7 là số may mắn vì
  $7^2 = 49$
  $4^2 + 9^2 = 97$
  $9^2 + 7^2 = 130$
  $1^2 + 3^2 + 0^2 = 10$
  $1^2 + 0^2 = 1$
- Những số may mắn dưới 500 là: 1, 7, 10, 13, 19, 23, 28, 31, 32, 44, 49, 68, 70, 79, 82, 86, 91, 94, 97, 100, 103, 109, 129, 130, 133, 139, 167, 176, 188, 190, 192, 193, 203, 208, 219, 226, 230, 236, 239, 262, 263, 280, 291, 293, 301, 302, 310, 313, 319, 320, 326, 329, 331, 338, 356, 362, 365, 367, 368, 376, 379, 383, 386, 391, 392, 397, 404, 409, 440, 446, 464, 469, 478, 487, 490, 496.

Câu 3: (*3 điểm*)

3.1. Đưa dãy số sau vào minHeap: `98,87,76,65,55,43,32,21,10`.

3.2. Vẽ lại Heap sau khi xóa xong số `55`.

- HẾT –

ĐỀ KIỂM TRA GIỮA KỲ MÔN: CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT Thời gian: **60 phút** (**KHÔNG** sử dụng tài liệu)	36

Câu 1: (*3 điểm*) Viết chương trình minh họa cho những số <100 về giả thuyết của nhà toán học người Đức - *Christian Goldbach* (1690-1764, trong một bức thư ngày 07/6/1742 gửi cho nhà toán học tài ba *Leonhard Euler*): ***mọi số tự nhiên chẵn lớn hơn 2 đều có thể viết được thành tổng của hai số nguyên tố*** (giả thuyết này đã được chỉ ra là đúng tới 4×10^{18}).

Ví dụ `4 = 2 + 2, 8 = 5 + 3, 20 = 13 + 7`

Câu 2: (*4 điểm*) Cho khai báo cấu trúc như sau:

```
struct TNODE
{   int Key;
    struct TNODE *pLeft, *pRight;
};
```

Giả sử cây nhị phân tìm kiếm đã được tạo trước đó. Viết hàm đếm tất cả các node có chứa *Key* đều là số Palindrome.

> 🕮 ***Số Palindrome (hoặc Palindromic)***
>
> • (Theo Wikipedia) Là một số vẫn giữ nguyên giá trị khi các chữ số của nó được đảo ngược. Hay nói cách khác là số đối xứng.
>
> • 30 số thập phân *palindrome* đầu tiên là: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 22, 33, 44, 55, 66, 77, 88, 99, 101 , 111, 121, 131, 141, 151, 161, 171, 181, 191, 202, ...

Câu 3: (*3 điểm*)

3.1. Đưa dãy số sau vào maxHeap: `12, 23, 34, 45, 56, 67, 78, 89, 90`.

3.2. Vẽ lại Heap sau khi xóa xong số `56`.

- HẾT –