Tin học cơ sở 4

Buổi 6. Các thao tác với mảng

Bộ môn Khoa học máy tính - 2017

Nội dung buổi học

1. Tìm kiếm, sửa các phần tử
2. Chèn, xóa một, nhiều phần tử
3. Sắp xếp
4. Mảng 2 chiều

Quản lý dữ liệu

- Nảy sinh nhu cầu khi có nhiều dữ liệu
  - Tìm kiếm (search)
  - Sửa, cập nhật (update)
  - Chèn thêm, xóa đi (insert, delete)
  - Sắp xếp (sort)

Tìm kiếm

- Tìm phần tử thỏa mãn điều kiện
  - Tìm các số lớn hơn 4
  - Tìm vị trí các số nguyên tố
  - Đếm số lượng số chẵn
  - …

Tìm kiếm

- Đoạn mã tìm kiếm mẫu

```
for (int i = 0; i < n; i++) {
  if ( isConditionSatisfied(a[i]) ) {
    // đoạn mã xử lý khi
    // a[i] thỏa mãn điều kiện
    ...
  }
}
```

CT1. Tìm các số lớn hơn 4

```
for (int i = 0; i < n; i++) {
   if ( a[i] > 4 ) {
      cout << "a[" << i << "] = "
           << a[i] << endl;
   }
}
```

4

7

0

6

8

1

CT2. Tìm các số nguyên tố

```
for (int i = 0; i < n; i++) {
  if ( isPrime(a[i]) ) {
    cout << "a[" << i << "] = "
         << a[i] << endl;
  }
}
```

```
bool isPrime(int num) {
  if (n <= 1) return false;
  for (int i = 2; i < num; i++) {
    if (num % i == 0)
      return false;
  }
  return true;
}
```

4

7

0

6

8

1

CT3. Đếm số lượng số chẵn

```
int count = 0;
for (int i = 0; i < n; i++) {
   if (a[i] % 2 == 0)
      count++;
}
cout << count << endl;
```

4

7

0

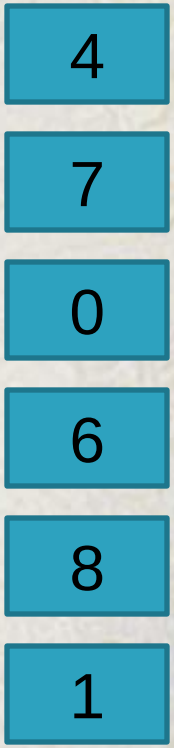
6

8

1

Sửa, cập nhật phần tử

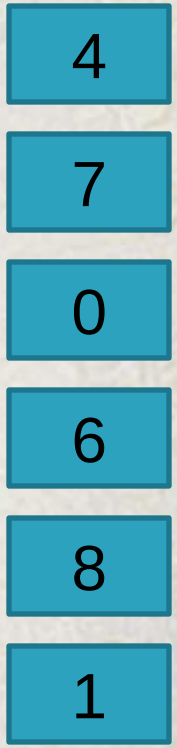
- Cập nhật phần tử
  - Khi phần tử thỏa mãn điều kiện
  - Bằng hàm, công thức tính từ các phần tử khác
  - …



Sửa, cập nhật các phần tử

- Đoạn mã cập nhật mẫu

```
for (int i = 0; i < n; i++) {
  if ( isConditionSatisfied(a[i]) ) {
    // đoạn mã cập nhật khi
    // a[i] thỏa mãn điều kiện
    ...
  }
}
```



CT4. Sửa các số lớn hơn 4 thành 4

```
for (int i = 0; i < n; i++) {
   if ( a[i] > 4 ) {
      a[i] = 4;
   }
}
```

4

7

0

6

8

1

CT4. Sửa các số lớn hơn 4 thành 4

```
for (int i = 0; i < n; i++) {
   if ( a[i] > 4 ) {
      a[i] = 4;
   }
}
```

CT5. Sửa các số thành tổng của nó và phần tử phía trước

```
for (int i = 0; i < n; i++) {
  a[i] += (i > 0 ? a[i-1] : 0);
}
```

Chỉ số l
hợp lệ k

Biểu thức lựa chọn ? :

4

7

0

6

8

1

CT5. Sửa các số thành tổng của nó và phần tử phía trước

```
for (int i = 0; i < n; i++) {
  a[i] += (i > 0 ? a[i-1] : 0);
}
```

4

11

11

17

25

26

Do vòng lặp for: các phần tử bị sửa trước khi cộng

CT5. Sửa các số thành tổng của nó và phần tử phía trước

- Kỹ thuật lặp ngược lại

```
for (int i = n-1; i >= 0; i--) {
   a[i] += (i > 0 ? a[i-1] : 0);
}
```

Luôn cộng các phần tử chưa bị sửa

4
7
0
6
8
1

CT5. Sửa các số thành tổng của nó và phần tử phía trước

- Kỹ thuật lặp ngược lại

```
for (int i = n-1; i >= 0; i--) {
   a[i] += (i > 0 ? a[i-1] : 0);
}
```

Luôn cộng các phần tử chưa bị sửa

4
11
7
6
14
9

CT6. Sửa các số thành tổng hai phần tử bên cạnh

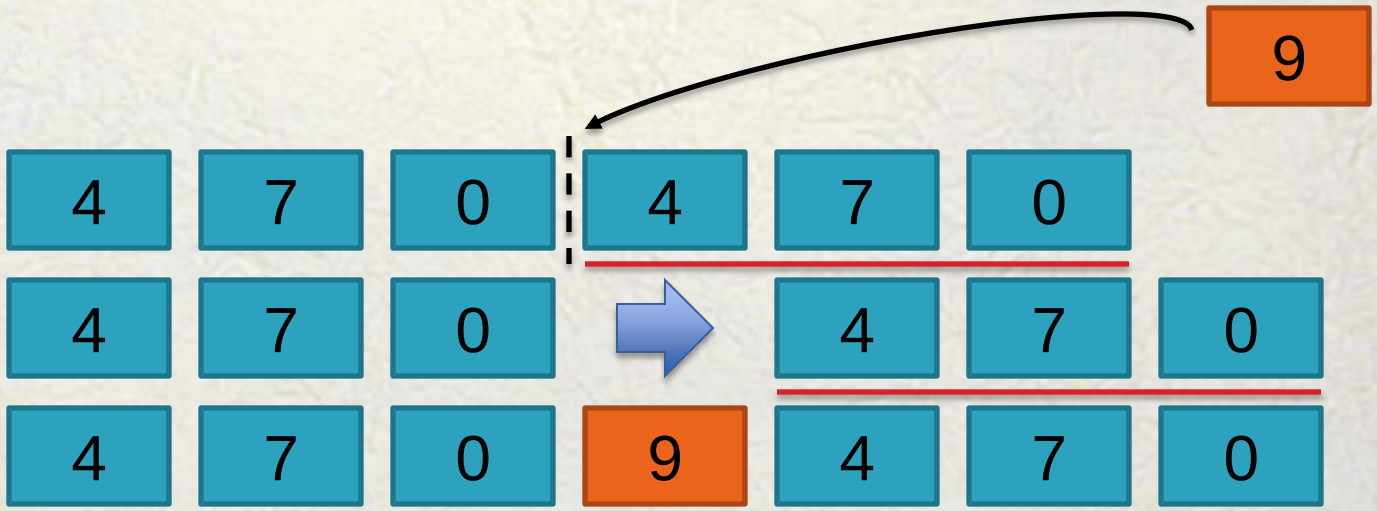
- Kỹ thuật lưu phần tử trước khi thao tác

```
int prev = 0;
for (int i = 0; i < n; i++) {
  int realPrev = prev;
  prev = a[i];
  a[i] = (i > 0 ? realPrev : 0)
       + (i < n-1 ? a[i+1] : 0);
}
```

Lưu trữ a[i-1]

Chèn thêm phần tử

1. Đẩy các phần tử sang phải
2. Đưa phần tử cần chèn vào chỗ trống

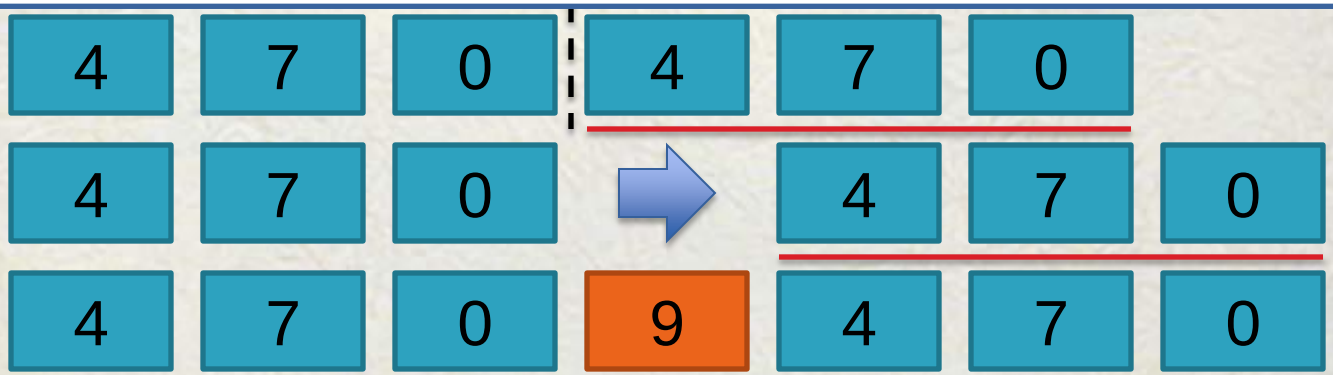


Chèn thêm phần tử

```
int insert(int arr[], int n, int pos, int value)
{
    for (int i = n; i > pos; i--) {
        arr[i] = arr[i-1];
    }
    arr[pos] = value;
    return n+1;
}
```

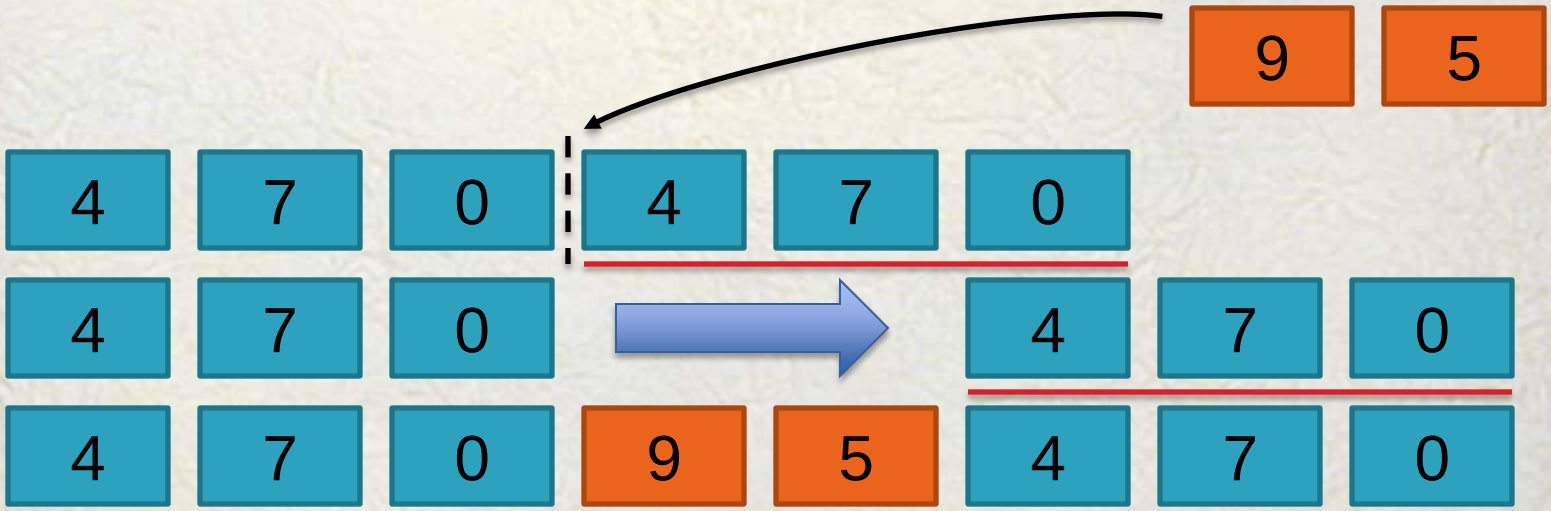
for ngược

Trả về số phần tử mới



Chèn mảng vào mảng

1. Đẩy các phần tử sang phải cho đủ chỗ
2. Đưa phần tử cần chèn vào chỗ trống

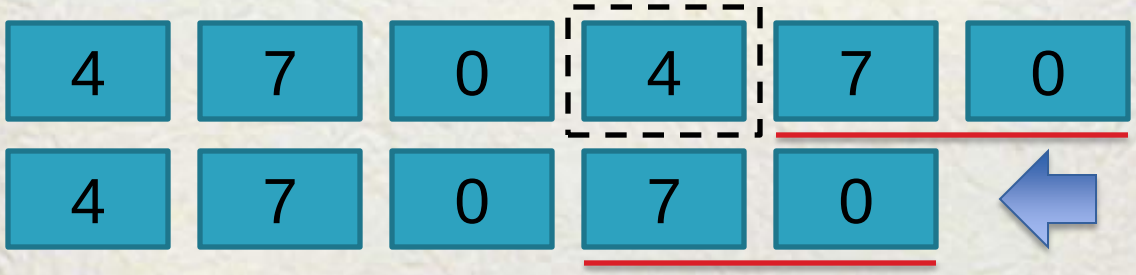


Chèn mảng vào mảng

```
int insertArrayToArray(int a[], int b[],
                       int na, int nb,
                       int pos)
{
   for (int i=na+nb-1; i>=nb+pos; i--)
     a[i]=a[i-nb];
   for (int i=0; i<nb; i++)
     a[i+pos]=b[i];
   return na+nb;
}
```

Xóa một phần tử

1. Đẩy các phần tử sang trái
2. Sửa lại số phần tử



Xóa một phần tử

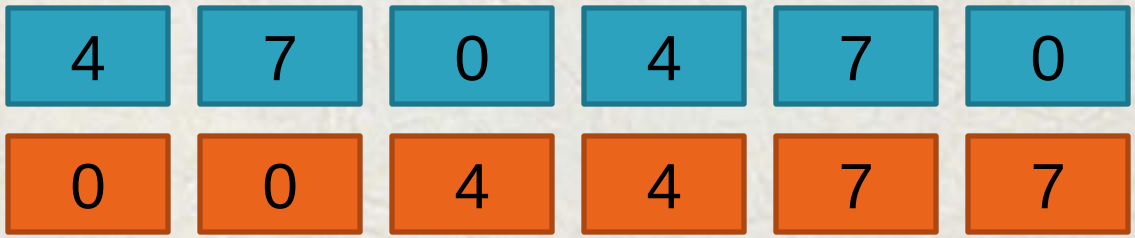
```
int remove(int arr[], int n, int pos)
{
    if (n == 0)
        return 0;

    for (int i = pos; i < n-1; i++) {
        arr[i] = arr[i+1];
    }
    return n-1;
}
```

for xuôi

Sắp xếp mảng

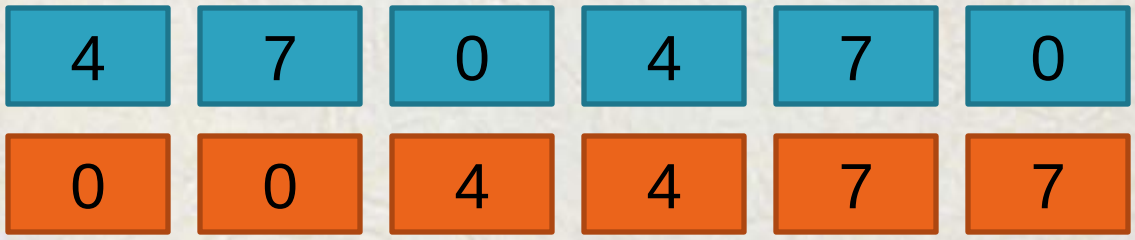
- Tăng dần
- Giảm dần
- Thứ tự từ điển



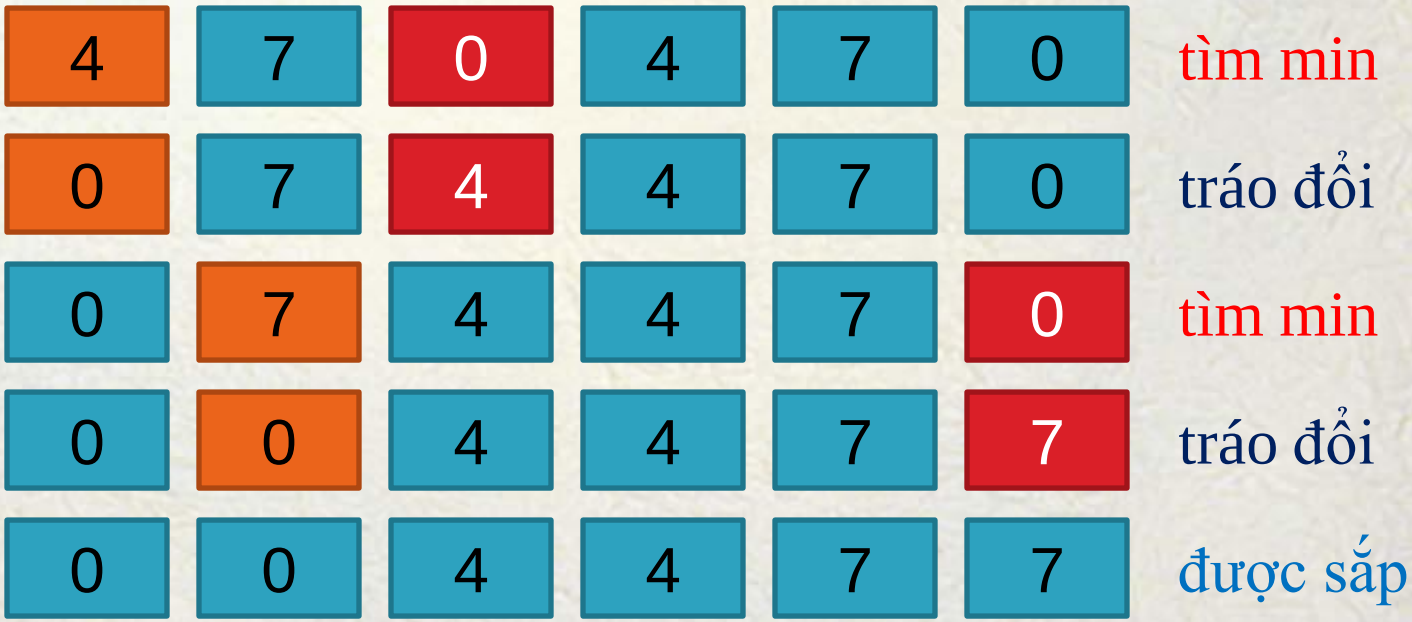
Sắp xếp mảng tăng dần

Thuật toán **Sắp xếp chọn** (selection sort)

1. Cho i đi từ 0 đến n-1 (trái qua phải)
2. Tìm vị trí phần tử nhỏ nhất trong a[i], a[i+1], …, a[n-1]
3. Tráo đổi phần tử đó với a[i]



Sắp xếp chọn


4 7 0 4 7 0 tìm min
0 7 4 4 7 0 tráo đổi
0 7 4 4 7 0 tìm min
0 0 4 4 7 7 tráo đổi
0 0 4 4 7 7 được sắp

Sắp xếp chọn

```
void sort(int arr[], int n)
{
    for (int i = 0; i < n; i++) {
        int minIndex = i;
        for (int j = i; j < n; j++) {
            if (a[j] < a[minIndex])
                minIndex = j;
        }
        int tmp = a[i];
        a[i] = a[minIndex];
        a[minIndex] = tmp;
    }
}
```

Biến thể của sắp xếp chọn

```
void sort(int arr[], int n)
{
    for (int i = 0; i < n; i++) {
        for (int j = i; j < n; j++) {
            if (a[j] < a[i]) {
                int tmp = a[i];
                a[i] = a[j];
                a[j] = tmp;
            }
        }
    }
}
```

Mảng nhiều chiều

- Khai báo bảng 100 dòng 1000 cột

```
int num[100][1000];
```

- Khai báo khối 3 chiều

```
int num3[10][10][10];
```

Mảng nhiều chiều

- Truy xuất sử dụng nhiều chỉ số

```
num[1][10]      // dòng 1, cột 10
num[i][j]       // dòng i, cột j
num3[2][5][8]   //
num3[i][j][k]   // 3 chỉ số
```

Câu hỏi

- Về nội dung buổi học ?
- Về nội dung khóa học ?
- Về cách tổ chức ?