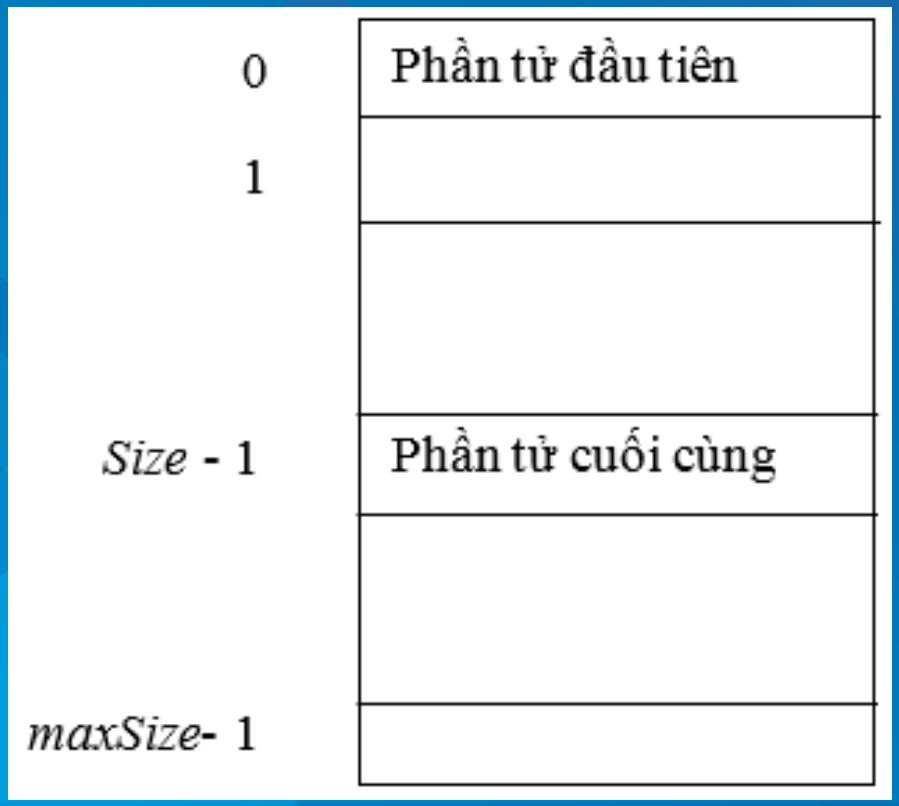
Cài đặt danh sách bởi mảng

- G/S độ dài tối đa của DS là ***maxSize***



Cài đặt danh sách bởi mảng

- Biểu diễn danh sách bởi cấu trúc gồm 2 trường

```
struct typeOfList
{
    typeOfItem elem[maxSize];
    int size;
}
typeOfList L;
```

Cài đặt danh sách bởi mảng- phép toán

- Khởi tạo danh sách rỗng

```
void initialize(typeOfList *L);
```

- Xác định độ dài của danh sách

```
int length(typeOfList L);
```

- Kiểm tra danh sách có đầy không

```
bool isFull(typeOfList L);
```

- Kiểm tra danh sách có rỗng không

```
bool isEmpty(TypeofList L);
```

Cài đặt danh sách bởi mảng- phép toán

- Loại phần tử có vị trí thứ v trong danh sách ra khỏi danh sách

```
void remove(int v, typeOfList *L);
```

- Xen phần tử x vào sau vị trí thứ v của danh sách

```
void insertAfter(int v, typeOfItem x, typeOfList *L);
```

- Tìm xem trong danh sách có chứa phần tử x hay không

```
int search(typeOfItem x, typeOfList L);
```

- Duyệt danh sách

```
void traverse(typeOfList L);
```

Cài đặt danh sách bởi mảng- phép toán

```
void  insertAfter(int v, typeOfItem x, typeOfList *L)
{
    int i;
    if (isEmpty(*L)==true)
    {
        L->size++;
         L->elem[L->size-1] = x;
    }
    else if(isFull(*L)==false)
    {
        L->size++;
        i=L->size-1;
        while (i>v+1)
        {
            L->elem[i]=L->elem[i-1];
            i--;
            printf("\n %d",i);
         }
         L->elem[i] = x;
    }
    else printf("\nDanh sach day");
}
```

Cài đặt danh sách bởi mảng- phép toán

```
void remove(int v, typeOfList *L)
{
    int i;
    if(isEmpty(*L)==false)
    {
        for (i=v; i<=L->size-1; i++)
        L->elem[i]=L->elem[i+1];
        L->size--;
    }
    else printf("\nDanh sach rong");
}
```

Tìm kiếm tuần tự trên danh sách

- Phương pháp:
  - lưu trữ các mẫu tin trong một mảng,
  - duyệt xuyên qua toàn bộ mảng một cách tuần tự

```
int  search(typeOfKey v, typeOfList L)
{
    int i;
    if(isEmpty(L)==false)
    {
        for(i=0; i<L.size; i++)
            if(v==L.elem[i].key)
            {
                printf("\n i = %d",i);
                return i;
            }
    }
    return -1;
}
```

Tìm kiếm nhị phân

```
int binarySearch(typeOfItem x, typeOfList L)
{
    int  m, bottom, top;
    bool  found=false;
    bottom=0;
    top=L.size-1;
    while (found ==false && bottom <= top)
    {
        m=(bottom+top+1)/2;
        if(x.key==L.elem[m].key)
        {
            found = true; return m;
        }
        else if (x.key<L.elem[m-1].key) top=m-1;
        else bottom=m+1;
    }
    return -1;
}
```