Chương 1
THUẬT TOÁN INSERTION SORT

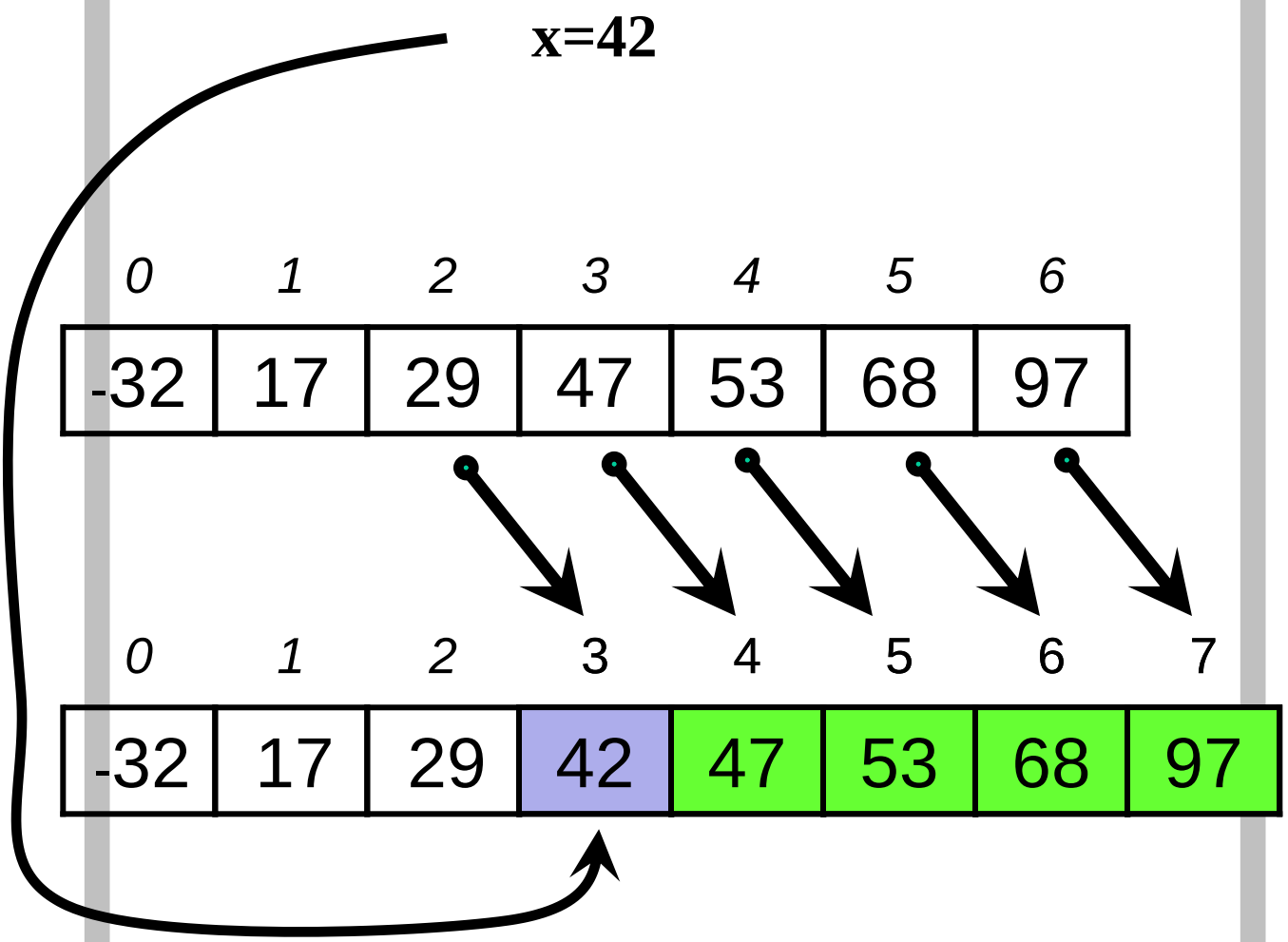
ThS. Nguyễn Tấn
Trần Minh Khang

1. BÀI TOÁN DẪN NHẬP

- Bài toán 1: Định nghĩa hàm thêm một giá trị x vào mảng một chiều các số thực có n phần tử đã được sắp tăng sao cho mảng vẫn được sắp tăng.

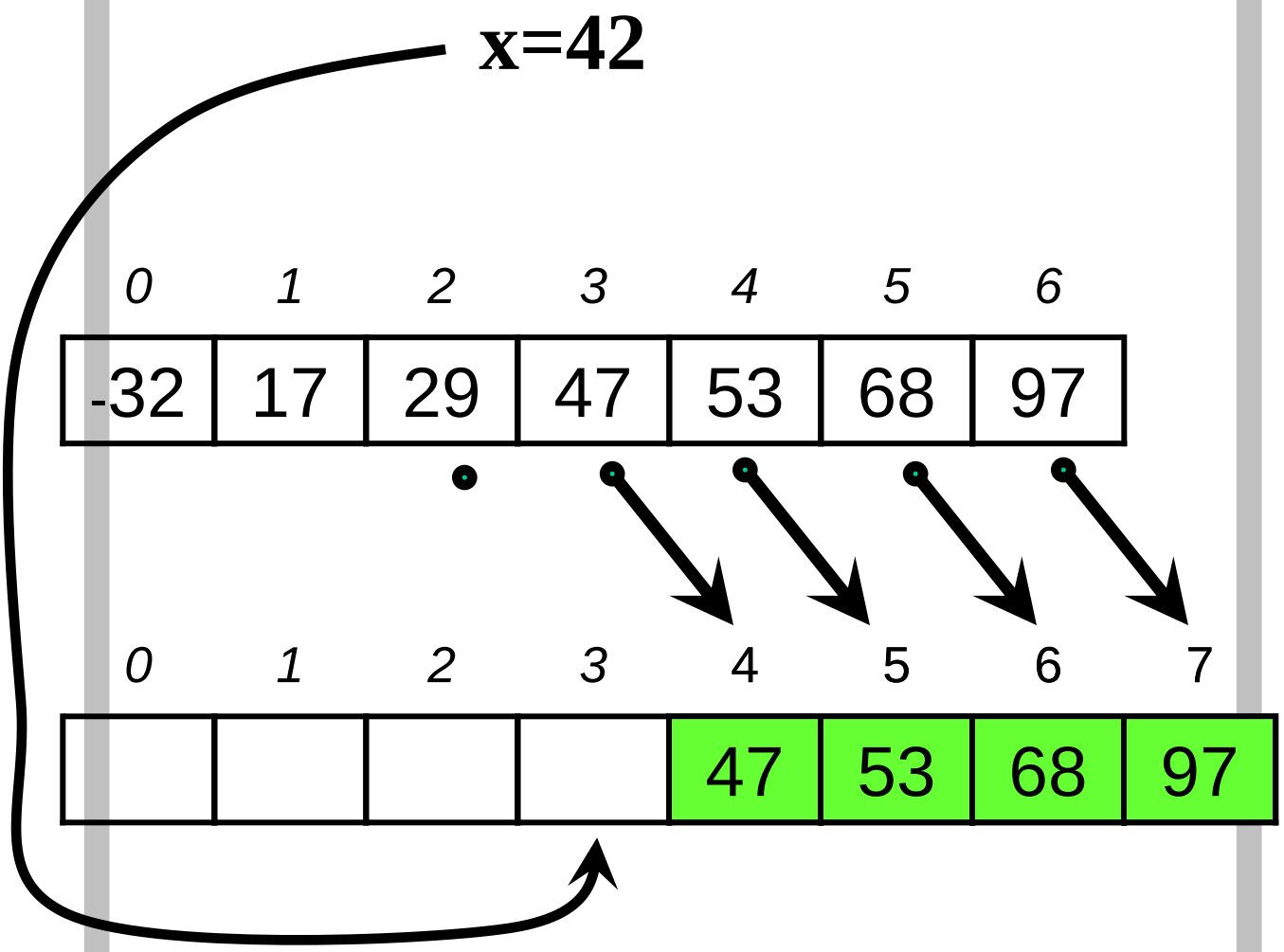
1. BÀI TOÁN DẪN NHẬP

- Ví dụ:



1. BÀI TOÁN DẪN NHẬP

- Ví dụ:



1. BÀI TOÁN DẪN NHẬP

- Định nghĩa hàm

```
11. void ThemBaoToan(float a[],
                     int &n,float x)
12. {
13.     int i = n-1;
14.     while(i>=0 && a[i]>x)
15.     {
16.         a[i+1] = a[i];
17.         i--;
18.     }
19.     a[i+1] = x;
20.     n++;
21. }
```

1. BÀI TOÁN DẪN NHẬP

- Định nghĩa hàm

```
11. void ThemBaoToan(float a[],
                     int &n,float x)
12. {
13.     int i = n-1;
14.     for(;(i>=0 && a[i]>x);)
15.     {
16.         a[i+1] = a[i];
17.         i--;
18.     }
19.     a[i+1] = x;
20.     n++;
21. }
```

1. BÀI TOÁN DẪN NHẬP

- Định nghĩa hàm

```
11. void ThemBaoToan(float a[],
                     int &n,float x)
12. {
13.     for(int i=n-1;
           (i>=0 && a[i]>x);)
14.     {
15.         a[i+1] = a[i];
16.         i--;
17.     }
18.     a[i+1] = x;
19.     n++;
20. }
```

1. BÀI TOÁN DẪN NHẬP

- Định nghĩa hàm

```
11. void ThemBaoToan(float a[],
                     int &n,float x)
12. {
13.     for(int i=n-1;
            (i>=0 && a[i]>x);i--)
14.     {
15.         a[i+1] = a[i];
16.     }
17.     a[i+1] = x;
18.     n++;
19. }
```

1. BÀI TOÁN DẪN NHẬP

- Định nghĩa hàm

```
11. void ThemBaoToan(float a[],
                     int &n,float x)
12. {
13.     for(int i=n-1;
           (i>=0 && a[i]>x);i--)
14.         a[i+1] = a[i];
15.     a[i+1] = x;
16.     n++;
17. }
```

1. BÀI TOÁN DẪN NHẬP

- Định nghĩa hàm

```
11. void ThemBaoToan(float a[],
                     int &n,float x)
12. {
13.     for(int j=n-1;
           (j>=0 && a[j]>x);j--)
14.         a[j+1] = a[j];
15.     a[j+1] = x;
16.     n++;
17. }
```

1. BÀI TOÁN DẪN NHẬP

- Bài toán 2: Định nghĩa hàm thêm một giá trị x vào mảng một chiều các số thực có i phần tử đã được sắp tăng sao cho mảng vẫn được sắp tăng.

1. BÀI TOÁN DẪN NHẬP

- Định nghĩa hàm

```
11. void ThemBaoToan(float a[],
                     int &i,float x)
12. {
13.     for(int j=i-1;
           (j>=0 && a[j]>x);j--)
14.         a[j+1] = a[j];
15.     a[j+1] = x;
16.     i++;
17. }
```

2. TƯ TƯỞNG THUẬT TOÁN

- Thuật toán Insertion sort sắp xếp dựa trên tư tưởng là không gian cần sắp xếp đã được sắp xếp một phần và ta chỉ cần thêm một giá trị mới vào không gian này sao cho không gian mới được sắp xếp mà thôi.

3. THUẬT TOÁN INSERTION SORT

- Bước 1: Chèn a[1] vào mảng a có 1 phần tử đã được sắp tăng để được mảng a có 2 phần tử sắp tăng.
- Bước 2: Chèn a[2] vào mảng a có 2 phần tử đã được sắp tăng để được mảng a có 3 phần tử sắp tăng.
- …
- Bước i: Chèn a[i] vào mảng a có i phần tử đã được sắp tăng để được mảng a có (i+1) phần tử sắp tăng.
- …
- Bước n-1: Chèn a[n-1] vào mảng a có (n-1) phần tử đã được sắp tăng để được mảng a có n phần tử sắp tăng.

4. HÀM CÀI ĐẶT

- Bài toán: Định nghĩa hàm sắp mảng một chiều các số nguyên tăng dần bằng thuật toán Insertion sort.

4. HÀM CÀI ĐẶT

- Bài toán: Định nghĩa hàm sắp mảng một chiều các số nguyên tăng dần bằng thuật toán Insertion sort.
- Hàm cài đặt

```
11. void InsertionSort(int a[],
                       int n)
12. {
13.     int i = 1;
14.     while(i<=n-1)
15.     {
16.         int x = a[i];
17.         for(int j=i-1;
                j>=0&&a[j]>x;j--)
18.             a[j+1] = a[j];
19.         a[j+1] = x;
20.         i++;
21.     }
22. }
```

4. HÀM CÀI ĐẶT

- Cải tiến hàm cài đặt

```
11. void InsertionSort(int a[],
                       int n)
12. {
13.     int i = 1;
14.     for(         ;i<=n-1;    )
15.     {
16.         int x = a[i];
17.         for(int j=i-1;
                j>=0&&a[j]>x;j--)
18.             a[j+1]=a[j];
19.         a[j+1] = x;
20.         i++;
21.     }
22. }
```

4. HÀM CÀI ĐẶT

- Cải tiến hàm cài đặt

```
11. void InsertionSort(int a[],
                       int n)
12. {
13. |
14. |    for(int i=1;i<=n-1;    )
15. |    {
16. |    |    int x = a[i];
17. |    |    for(int j=i-1;
    |    |       j>=0&&a[j]>x;j--)
18. |    |        a[j+1]=a[j];
19. |    |    a[j+1] = x;
20. |    |    i++;
21. |    }
22. }
```

4. HÀM CÀI ĐẶT

- Cải tiến hàm cài đặt

```
11. void InsertionSort(int a[],
                       int n)
12. {
13. |
14. |    for(int i=1;i<=n-1;i++)
15. |    {
16. |    |    int x = a[i];
17. |    |    for(int j=i-1;
    |    |       j>=0&&a[j]>x;j--)
18. |    |        a[j+1]=a[j];
19. |    |    a[j+1] = x;
    |    |
21. |    }
22. }
```

5. DANH SÁCH LIÊN KẾT KÉP

- Bài toán: Định nghĩa hàm sắp dslk kép các số nguyên tăng dần bằng thuật toán Insertion sort.
- Cấu trúc dữ liệu

```
11. struct node
12. {
13.     int info;
14.     struct node*pNext;
15.     struct node*pPrev;
16. };
17. typedef struct node NODE;
18. struct list
19. {
20.     NODE*pHead;
21.     NODE*pTail;
22. };
23. typedef struct list LIST;
```

5. DANH SÁCH LIÊN KẾT KÉP

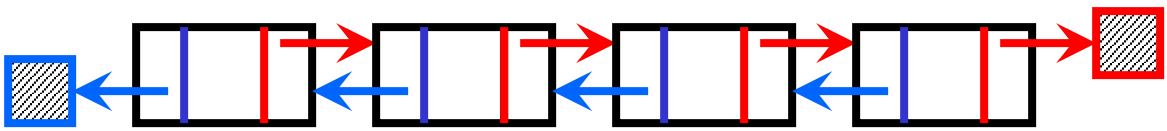
- Cải tiến hàm cài đặt

```
11. void InsertionSort(int a[],
                       int n)
12. {
13.
14.     for(int i=1;i<=n-1;i++)
15.     {
16.         int x = a[i];
17.         for(int j=i-1;
                j>=0&&a[j]>x;j--)
18.             a[j+1]=a[j];
19.         a[j+1] = x;
21.     }
22. }
```

5. DANH SÁCH LIÊN KẾT KÉP

```
11. void InsertionSort(LIST &ℓ)
12. {
13.    for(NODE*p=ℓ.pHead->pNext;p;
                                  p=p->pNext)
14.    {
15.       int x = p->info;
16.       for(NODE*q=p->pPrev;
             q&&q->info>x;q=q->pPrev)
17.          q->pNext->info=q->info;
18.       q->pNext->info = x;
19.    }
20. }
```

5. DANH SÁCH LIÊN KẾT KÉP



5. DANH SÁCH LIÊN KẾT KÉP

```
11. void InsertionSort(LIST &ℓ)
12. {
13.   for(NODE*p=ℓ.pHead->pNext;p;
                              p=p->pNext)
14.   {
15.     int x = p->info;
16.     for(NODE*q=p->pPrev;
           q&&q->info>x;q=q->pPrev)
17.       q->pNext->info=q->info;
18.     if(q)
19.       q->pNext->info = x;
20.     else
21.       ℓ.pHead->info = x;
22.   }
23. }
```

5. DANH SÁCH LIÊN KẾT KÉP

- Bài toán: Hãy sắp xếp danh sách liên kết kép tọa độ các điểm trong không gian tăng dần theo khoảng cách đến góc tọa độ.

- Khai báo cấu trúc dữ liệu

```
10. struct diemkg
11. {
12.     float x;
13.     float y;
14.     float z;
15. };
16. typedef struct diemkg DIEMKG;
17. struct node
18. {
19.     DIEMKG info;
20.     struct node*pNext;
21.     struct node*pPrev;
22. };
23. typedef struct node NODE;
24. struct list
25. {
26.     NODE*pHead;
27.     NODE*pTail;
28. };
29. typedef struct list LIST;
```

- Định nghĩa hàm

```
1. float KhoangCachGoc(DIEMKG P)
2. {
3.     return sqrt(P.x*P.x+
                   P.y*P.y+
                   P.z*P.z);
4. }
```

- Bài toán: Hãy sắp xếp danh sách liên kết kép tọa độ các điểm trong không gian tăng dần theo khoảng cách đến góc tọa độ.

- Hàm cài đặt

```
11. void InsertionSort(LIST &ℓ)
12. {
13.   for(NODE*p=ℓ.pHead->pNext;p;
                              p=p->pNext)
14.   {
15.     DIEMKG temp = p->info;
16.     for(NODE*q=p->pPrev;
            q &&
            KhoangCachGoc(q->info)
            >KhoangCachGoc(temp);
            q=q->pPrev)
17.       q->pNext->info=q->info;
18.     if(q)
19.       q->pNext->info = temp;
20.     else
21.       ℓ.pHead->info = temp;
22.   }
23. }
```