Chương 10
Bộ nhớ động và ứng dụng trong danh sách liên kết

- Đặt vấn đề
- Các hàm cấp phát bộ nhớ động
- Khai báo và sử dụng danh sách liên kết đơn
- Minh họa

VÙNG NHỚ

Stack, heap

Stack: Vùng nhớ cần thiết được cấp phát khi gọi hàm.
Tự động giải phóng vùng nhớ sau khi kết thúc gọi hàm.

Heap: Vùng nhớ mang tính toàn cục.
Vùng nhớ được đề nghị cấp phát bởi việc gọi hàm cấp phát.
Vùng nhớ được đề nghị giải phóng bởi việc gọi hàm giải phóng.

CÁC HÀM KHAI THÁC BỘ NHỚ ĐỘNG

Cấp phát

```
1   void *malloc(int dodaibyte);
2
3
4
5
6
7
8
9
10  ==================
```

Cấp phát ***một vùng bộ nhớ*** từ **heap**.
Số byte là **dodaibyte**.

Giá trị của hàm là địa chỉ vùng nhớ được cấp phát
Giá trị của hàm là NULL khi không có đủ vùng nhớ để cấp phát.

CÁC HÀM KHAI THÁC BỘ NHỚ ĐỘNG

Cấp phát

```
1   void *malloc(int dodaibyte);
2
3   void *calloc(int soptu, int dodaibyteptu);
4
5
6
7
8
9
10  ==================
```

Cấp phát ***bộ nhớ*** từ **heap**. Khởi tạo 0 cho tất cả các byte trong vùng nhớ.
Số lượng phần tử được cấp phát là **soptu**
Mỗi phần tử có kích thước là **dodaibyteptu**.

Giá trị của hàm là địa chỉ vùng nhớ được cấp phát
Giá trị của hàm là NULL khi không có đủ vùng nhớ để cấp phát.

CÁC HÀM KHAI THÁC BỘ NHỚ ĐỘNG

Cấp phát

```
1   void *malloc(int dodaibyte);
2
3   void *calloc(int soptu, int dodaibyteptu);
4
5   void *realloc(void *ctro, int dodaibyte);
6
7
8
9
10  ==================
```

1 Cấp phát lại ***vùng bộ*** đã cấp phát từ **heap. ctro** là địa chỉ vùng nhớ cũ.
2 Số byte là **dodaibyte**.
3 Dữ liệu cũ nằm trong phạm vi **dodaibyte** được bảo toàn.
4
5 Giá trị của hàm là địa chỉ vùng nhớ được cấp phát
6 Giá trị của hàm là NULL khi không có đủ vùng nhớ để cấp phát.
7
8
9
20

CÁC HÀM KHAI THÁC BỘ NHỚ ĐỘNG

Cấp phát

```
1   void *malloc(int dodaibyte);
2
3   void *calloc(int soptu, int dodaibyteptu);
4
5   void *realloc(void *ctro, int dodaibyte);
6
7   void free(void *ctro);
8
9
10  ==================
```

Giải phóng ***vùng nhớ*** cấp phát có địa chỉ là **ctro**.

Giá trị **ctro** phải là địa chỉ vùng nhớ hợp lệ từ **heap**

MINH HỌA

Biết số phần tử trước khi cấp phát

```
1   int *nhapmangp(int *n) {
2       int *tam, i;
3       printf("Nhap so ptu day so nguyen: ");scanf("%d",n);
4       if(*n<=0) return NULL;
5       tam = (int *)calloc(*n,sizeof(int));
6       printf("Nhap ptu day:");
7       if(tam){
8           for(i=0;i<*n;i++) scanf("%d",&tam[i]);
9       }
10      return tam;
1   }
2
3   void xuatmang(int a[], int n) {
4       int i;
5       for(i=0;i<n;i++) printf("%d",a[i]);
6   }
7
8
9
20
```

MINH HỌA

Biết số phần tử trước khi cấp phát

```
1   #include <malloc.h>
2   #include <stdio.h>
3
4   void xuatmang(int [], int);
5   void *nhapmangp(int *);
6
7   void main()
8   {
9       int *a, n;
10      a = nhapmangp(&n);
1       if(a) {
2           xuatmang(a,n);
3           free(a);
4       }
5   }
6
7
8
9
20
```

MINH HỌA

Không cho biết số phần tử trước khi cấp phát

```
1   int *themptu(int *day, int *n, int ptu)
2   {   int *tam;
3       tam = (int *)realloc(day,++(*n)*sizeof(int));
4       if(tam) tam[*n-1] = ptu;
5       return tam;
6   }
7   int *nhapmang(int *n)
8   {   int *tam, *day = NULL, ptu;
9      *n = 0;
10      while (scanf("%d",&ptu)) {
1          tam = themptu(day,n,ptu);
2          if (tam) day = tam;
3      }
4      return day;
5   }
6   void xuatmang(int a[], int n)
7   {   int i;
8       for(i=0;i<n;i++) printf("%d",a[i]);
9   }
20
```

MINH HỌA

Không cho biết số phần tử trước khi cấp phát

```
1    #include <malloc.h>
2    #include <stdio.h>
3
4    void xuatmang(int [], int);
5    int *nhapmang(int *);
6    int *themptu(int *, int *, int);
7
8    void main()
9    {
10       int *a, n;
1        printf("Nhap cac ptu day, ket thuc bang Ctrl+x");
2        a = nhapmang(&n);
3        if (a) {
4           xuatmang(a,n);
5           free(a);
6        }
7    }
8
9
20
```

DANH SÁCH LIÊN KẾT ĐƠN

Khai báo

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   //====================
4   typedef struct nut *lket;
5   struct nut {
6               int gtri;
7               lket ktiep;
8   };
9   //====================
10
1
2
3
4
5
6
7
8
9
20
```

DANH SÁCH LIÊN KẾT ĐƠN

Xây dựng

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   //====================
4   typedef struct nut *lket;
5   struct nut {
6               int gtri;
7               lket ktiep;
8   };
9   //====================
10  lket taoNutDon(int k) {
1               lket tam;
2               tam = (lket) malloc(sizeof(struct nut));
3               if (!tam){
4                       printf("cap phat nutDon that bai!");exit(0);
5               }
6               tam->gtri = k;
7               tam->ktiep = NULL;
8               return tam;
9   }
20
```

DANH SÁCH LIÊN KẾT ĐƠN

Xây dựng

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   //====================
4   typedef struct nut *lket;
5   struct nut {
6               int gtri;
7               lket ktiep;
8   };
9   //====================
10  lket themNutDau(lket *dau, lket *cuoi, int k) {
1               lket nuttam = taoNutDon(k);
2               if (!*dau)
3                       *dau = *cuoi = nuttam;
4               else {
5                       nuttam->ktiep = *dau;
6                       *dau = nuttam;
7               }
8   }
9
20
```

DANH SÁCH LIÊN KẾT ĐƠN

Xây dựng

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   //====================
4   typedef struct nut *lket;
5   struct nut {
6               int gtri;
7               lket ktiep;
8   };
9   //====================
10  lket themNutCuoi(lket *dau, lket *cuoi, int k) {
1               lket nuttam = taoNutDon(k);
2               if (!*dau)
3                       *dau = *cuoi = nuttam;
4               else {
5                       (*cuoi)->ktiep = nuttam;
6                       *cuoi = nuttam;
7               }
8   }
9
20
```

DANH SÁCH LIÊN KẾT ĐƠN

Luyện tập

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   //====================
4   typedef struct nut *lket;
5   struct nut {
6               int gtri;
7               lket ktiep;
8   };
9   //====================
10  void chenNutSau(lket nutmoc, int k);
1
2   void chenNutTruocp(lket dau, lket nutmoc, int k);
3
4   void xoaNutGiua(lket dau, lket bo);
5
6   void xuatDanhSach(lket dau);
7
8
9
20
```