第一次作业

第一题

1. 计算$\sum_{k=1}^{n} k!$，复杂度$O(n^2)$

```
int sum(int n) {
    int s = 0;
    for (int i = 1; i <= n; i++) {
        int p = 1;
        for (int j = 1; j <= i; j++)  //计算i!
            p *= j;
        s += p;
    }
    return s;
}
```

2. 计算$\sum_{k=1}^{n} k!$，复杂度$O(n)$

```
int fac(int n) {
    int p = 1, s = 0;
    for (int i = 1; i <= n; i++) {
        p *= i;   //利用上个循环计算i!
        s += p;
    }
    return s;
}
```

第二题

1. 复杂度$O(\sqrt{n})$

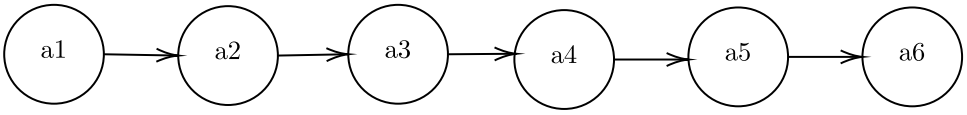
```
void f(int n) {
    int s, i = 0;
    while (s <= n) {//s~i^2<n
        ++i;
        s += i;   //计算1到i的和
        printf("*");
    }
}
```

2. 复杂度$O(n(logn)^2)$

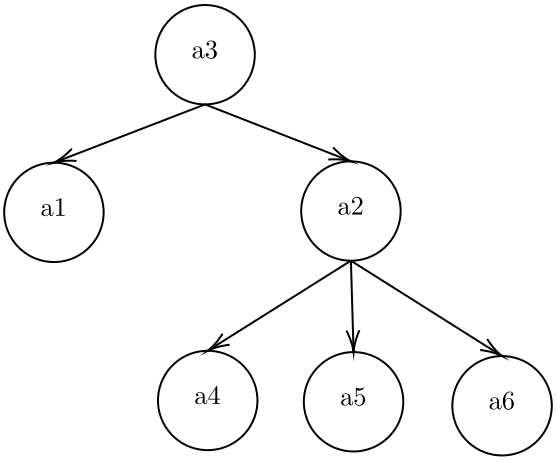
```
void f(int n) {
    int i, j, k, count = 0;
    for (i = n / 2; i <= n; i++)          // n/2次
        for (j = 1; j < n; j *= 2)        // logn次
            for (k = 1; k <= n; k *= 2)   // logn次
                count++;
}
```

第三题

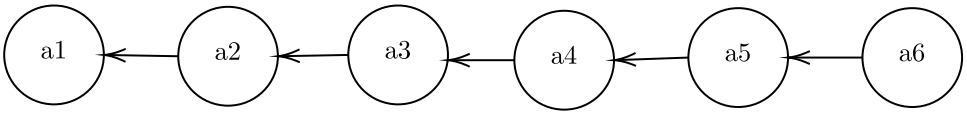
1. 线性结构



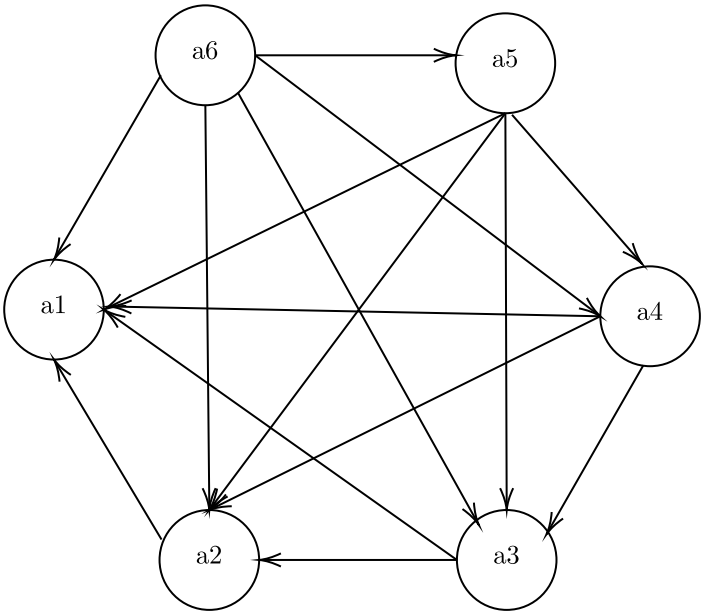
2. 树结构



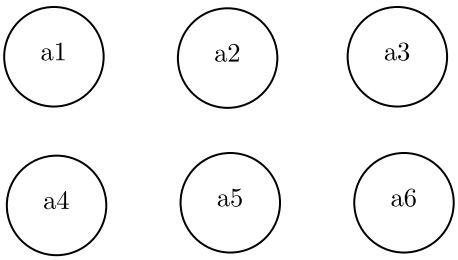
3. 线性结构



4. 图结构



5. 集合



第四题

复杂度$O(n)$ (长度为偶数时返回靠后的那个)

```
struct SNode *FindMiddle(LinkList &HA) {
    struct node *h = HA.head;
    int length = 0;
    if (h = nullptr)
        return nullptr;
    while (h- != nullptr) {
        length++;
        h = h->next;
    }
    length /= 2;
    h = HA.head;
    while (length-- > 0) {
        h = h->next;
    }
    return h;
}
```

第五题

1. 带表头

```
void mergeList(LinkList &HA, LinkList &HB, LinkList &HC) {
    struct SNode *i = HA.head->next;
    struct SNode *j = HB.head->next;
    struct SNode *s = HC.head;
    while (i != nullptr && j != nullptr) {
        struct SNode *n = new SNode;

        if (i->data <= j->data) {
            n->data = i->data;
            struct SNode *t = i;
            i = i->next;
            delete t;
        } else {
            n->data = j->data;
            struct SNode *t = j;
            j = j->next;
            delete t;
        }
        s->next = n;
        s = s->next;
    }
    if (i == nullptr) {
        while (j != nullptr) {
            struct SNode *n = new SNode;
            n->data = j->data;
            struct SNode *t = j;
            j = j->next;
            delete t;
            s->next = n;
            s = s->next;
        }
    } else {
        while (i != nullptr) {
            struct SNode *n = new SNode;
            n->data = i->data;
            struct SNode *t = i;
            i = i->next;
            delete t;
            s->next = n;
            s = s->next;
        }
    }
}
```

2. 不带表头

```
void mergeList(LinkList &HA, LinkList &HB, LinkList &HC) {
    struct SNode *i = HA.head;
    struct SNode *j = HB.head;
    struct SNode *s = HC.head;
    while (i != nullptr && j != nullptr) {
        struct SNode *n = new SNode;

        if (i->data <= j->data) {
            n->data = i->data;
            struct SNode *t = i;
            i = i->next;
            delete t;
        } else {
            n->data = j->data;
            struct SNode *t = j;
            j = j->next;
            delete t;
        }
        if (s == nullptr)
            HC.head = n;
        else {
            s->next = n;
            s = s->next;
        }
    }
    if (i == nullptr) {
        while (j != nullptr) {
            struct SNode *n = new SNode;
            n->data = j->data;
            struct SNode *t = j;
            j = j->next;
            delete t;
            if (s == nullptr)
                HC.head = n;
            else {
                s->next = n;
                s = s->next;
            }
        }
    } else {
        while (i != nullptr) {
            struct SNode *n = new SNode;
            n->data = i->data;
            struct SNode *t = i;
            i = i->next;
            delete t;
            if (s == nullptr)
                HC.head = n;
            else {
                s->next = n;
                s = s->next;
            }
        }
```

```
    }
}
```

第六题

1.

```
void inverList(SeqList &L) {
    int n = L.length;
    for (int i = 0; i < n / 2; i++) {
        int t = L.list[i];
        L.list[i] = L.list[n - i - 1];
        L.list[n - i - 1] = t;
    }
}
```

2.

```
void invertLinkList(LinkList &HL) {
    struct SNode *h = HL.head;
    struct SNode *prev = nullptr;
    while (h != nullptr) {
        struct SNode *next = curr->next;
        curr->next = prev;
        prev = curr;
        curr = next;
    }
}
```

第七题

XYZ
XZY
YXZ
YZX
ZYX

第八题

```
NODE *list_find(NODE *current, ElemType x) {
    if (current == nullptr)
        return nullptr;
    if (x != current->data)
        return list_find(current->next, x);
    return current;
}
```

第九题

功能：输出1到n

时间复杂度：$O(n)$

空间复杂度：O(n)

```
void writ(int n) {
    if (n != 0) {
        writ(n - 1);         //递归
        cout << n << endl;   //回溯
        return;
    }
}
```