



廣東工業大學

《程序设计实验》实验报告

学 院 先进制造学院

专 业 计算机科学与技术

年级班别 2022 级计算机科学与技术 8 班

学 号 3122008883

学生姓名 陈煜祺

指导教师 李泓澍

成 绩

广东工业大学

先进制造学院 计算机科学与技术专业 22(8)班 学号 3122008883

姓名 陈煜祺 协作者 无 教师评定 _____

实验题目 _____ **一、数字操作**

1、 题目内容

输入一个不多于 5 位的正整数。要求：求出它是几位数，分别输出每一位数字，并按逆序输出各位数字。（例如原数为 321，应输出 123）

输入用例：

123

输出用例：

位数：3
每位数字为：1，2，3
反序数字为：321

2、 题目分析与算法设计（分析思路、NS 图或伪代码）

用伪代码表示的算法如下：

```
begin

    input  $\Rightarrow$  n

    //模块 1：存储 n 的各位数字并对位数 cnt 进行计数
    while n>0{
        cnt++
        n%10 $\Rightarrow$ a[cnt]
        n/10 $\Rightarrow$ n
    }

    //模块 2：按题目要求输出处理后的数据
    print cnt           //输出位数
    for i=cnt to 1 :
        print a[i]      //分别输出每一位数字
    for i=1 to cnt :
        print a[i]      //输出反序数字

end
```

3、 代码实现

```
1  #include<stdio.h>
2
3  int n, cnt=0, a[6]={0};
4
5  int main()
6  {
7      scanf("%d", &n);
8      while(n>0)
9      {
10         cnt++;
11         a[cnt]=n%10;
12         n/=10;
13     }
14
15     printf("位数: %d\n", cnt);
16
17     printf("每位数字为: ");
18     for(int i=cnt; i>1; i--)
19     {
20         printf("%d, ", a[i]);
21     }
22     printf("%d\n", a[1]);
23
24     printf("反序数字为: ");
25     for(int i=1; i<=cnt; i++)
26     {
27         if(a[i])
28             printf("%d", a[i]);
29     }
30
31     return 0;
32 }
```

4、 实验结果（测试样例运行结果截图）

```
C:\Users\Administrator\Desktop\A.exe
123
位数: 3
每位数字为: 1, 2, 3
反序数字为: 321
Process returned 0 (0x0) execution time : 20.136 s
Press any key to continue.
```

5、 问题与总结（实验中遇到的难以解决的问题、对实验题目考察知识点的总结）

本次实验题目将“使用算术运算符对整型数字进行操作”作为主要的核心考察点，并附带考察了基本的输入输出和赋值操作等。实验要求我们掌握基本的上机操作能力，并着重考察了我们对于除法运算符“/”和取模运算符“%”的深入理解与熟练运用。

顺利完成本次实验题目的关键点在于程序算法的合理设计，特别是各位数字的储存和位数的计数，这是初学者难以解决的问题。同时也应注意代码书写习惯与缩进格式，在初学时养成有意识地提高代码可读性的良好习惯，能为将来进行更复杂的程序设计实验打下坚实的基础。

广东工业大学

先进制造学院 计算机科学与技术专业 22(8)班 学号 3122008883

姓名 陈煜祺 协作人 无 教师评定

实验题目 二、上下金字塔

1、 题目内容

题目描述

输出双层金字塔。

输入描述:

多个测试数据。每个测试数据输入一个整数 n ($2 \leq n \leq 9$)

输出描述:

输出双层金字塔

示例1

输入

2
5

输出

```
  *  
* * *  
  *  
    *  
  * * *  
* * * * *  
* * * * *  
* * * * *  
* * * * *  
  * * * *  
    * * *  
      *
```

2、 题目分析与算法设计（分析思路、NS 图或伪代码）

用伪代码表示的算法如下：

```
begin
    while input⇒n and n!=EOF{           //当仍有数据输入时继续执行程序

        //模块 1： 输出金字塔上半部分和中间一行
        for i=1 to n : {
            print ‘ ’    for (n-i)    times //对于第 i 行，应先输出 n-i 个空格
            print ‘*’    for (i*2-1) times //紧接着输出 i*2-1 个‘*’符号
            print ‘ ’    for (n-i)    times //紧接着再输出 n-i 个空格
            print ‘\n’                    //行末输出换行
        }

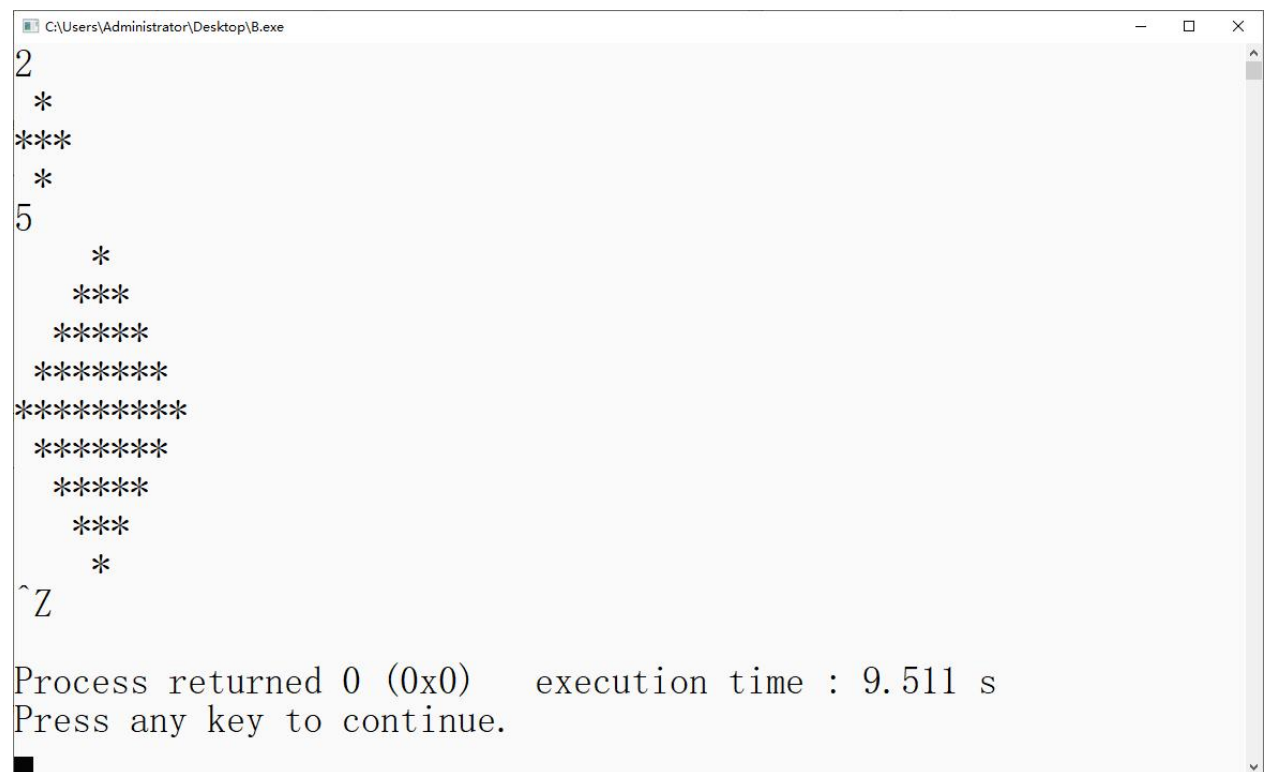
        //模块 2： 输出金字塔的下半部分（具体细节同理）
        for i=n-1 to 1: {
            print ‘ ’    for (n-i)    times
            print ‘*’    for (i*2-1) times
            print ‘ ’    for (n-i)    times
            print ‘\n’
        }

    }
end
```

3、 代码实现

```
1      #include<stdio.h>
2      int n;
3      int main()
4      {
5          while(scanf("%d",&n)!=EOF)
6          {
7              for(int i=1;i<=n;i++)
8              {
9                  for(int j=1;j<=n-i;j++)
10                     putchar(' ');
11
12                     for(int j=1;j<=(i<<1)-1;j++)
13                         putchar('*');
14
15                     for(int j=1;j<=n-i;j++)
16                         putchar(' ');
17
18                     putchar('\n');
19             }
20             for(int i=n-1;i>=1;i--)
21             {
22                 for(int j=1;j<=n-i;j++)
23                     putchar(' ');
24
25                     for(int j=1;j<=(i<<1)-1;j++)
26                         putchar('*');
27
28                     for(int j=1;j<=n-i;j++)
29                         putchar(' ');
30
31                     putchar('\n');
32             }
33         }
34         return 0;
35     }
```


4、 实验结果（测试样例运行结果截图）



```
C:\Users\Administrator\Desktop\B.exe
2
 *
***
 *
5
  *
 ***
*****
*****
*****
*****
 *****
  *****
   *****
    *
^Z
Process returned 0 (0x0)   execution time : 9.511 s
Press any key to continue.
```

5、 问题与总结（实验中遇到的难以解决的问题、对实验题目考察知识点的总结）

本次实验题目将循环结构及其嵌套作为核心考察点，并附带考察了简单的算法思维能力和检测文件尾（EOF）等操作。实验要求我们熟练掌握循环结构及其嵌套的书写，并着重考察了用循环结构实现简单算法的能力。

顺利完成本次实验题目的关键在于输出格式的精确控制，特别是空格和换行的处理。要顺利解决这个问题，需要仔细观察样例，找到输出样例中各符号的出现规律并认真归纳总结，从而找到用循环结构控制输出格式的方法，这是许多同学会遇到的难以解决的问题。

广东工业大学

先进制造学院 计算机科学与技术专业 22(8)班 学号 3122008883

姓名 陈煜祺 协作者 无 教师评定

实验题目 三、绝对值排序

1、 题目内容

A - 绝对值排序 HDU - 2020

输入 $n(n \leq 100)$ 个整数，按照绝对值从大到小排序后输出。题目保证对于每一个测试实例，所有的数的绝对值都不相等。

Input

输入数据有多组，每组占一行，每行的第一个数字为 n ，接着是 n 个整数， $n=0$ 表示输入数据的结束，不做处理。

Output

对于每个测试实例，输出排序后的结果，两个数之间用一个空格隔开。每个测试实例占一行。

Sample

Input	copy	Output	copy
3 3 -4 2 4 0 1 2 -3 0		-4 3 2 -3 2 1 0	

2、 题目分析与算法设计（分析思路、NS 图或伪代码）

用伪代码表示的算法如下：

```
begin
    //当 n 不为 0 时持续执行程序
    while (input $\Rightarrow$ n and n $\neq$ 0) {

        //模块 1：读入并存储待处理数据
        for i=1 to n :
            input $\Rightarrow$ a[i]

        //模块 2：对数组中的数据按绝对值大小进行降序冒泡排序
        for i=1 to n-1 :
            for j=1 to n-i :
                if abs(a[j]<a[j+1]) 交换 a[j]和 a[j+1]

        //模块 3：按题目格式要求输出排序好的数据
        for i=1 to n-1 :
            print a[i] + 空格
        print a[n] + 换行符
    }
end
```

3、 代码实现

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<math.h>
4
5  int n,temp,a[105];
6
7  int main()
8  {
9      while (scanf("%d",&n)&&n)
10     {
11         for(int i=1;i<=n;i++)
12         {
13             scanf("%d",&a[i]);
14         }
15         for(int i=1;i<=n-1;i++)
16         {
17             for(int j=1;j<=n-i;j++)
18             {
19                 if(abs(a[j])<abs(a[j+1]))
20                 {
21                     temp=a[j];
22                     a[j]=a[j+1];
23                     a[j+1]=temp;
24                 }
25             }
26         }
27         for(int i=1;i<=n-1;i++)
28         {
29             printf("%d ",a[i]);
30         }
31         printf("%d",a[n]);
32         printf("\n");
33     }
34
35     return 0;
36 }
```

4、 实验结果（测试样例运行结果截图）



```
C:\Users\Administrator\Desktop\A冒泡排序.exe
3 3 -4 2
-4 3 2
4 0 1 2 -3
-3 2 1 0
0

Process returned 0 (0x0)   execution time : 5.137 s
Press any key to continue.
■
```

5、 问题与总结（实验中遇到的难以解决的问题、对实验题目考察知识点的总结）

本次实验题目将冒泡排序算法和一维数组的使用作为核心考察点，并附带考察了循环嵌套书写和简单的变量交换等操作。实验要求我们熟练掌握一维数组的使用（读和写），并着重考察了用循环结构实现冒泡排序算法的能力。

顺利完成本次实验题目的关键在于如何将数组中的数据按绝对值大小进行降序冒泡排序。要顺利解决这个问题，需要我们熟练掌握冒泡排序算法的思想，并根据本次实验题目的要求，将课本中的冒泡排序原型进行调整，修改其中的判断条件以实现不同比较准则（如本次实验中要求按绝对值大小进行比较）的排序，从而完成实验题目的要求。

广东工业大学

先进制造学院 计算机科学与技术专业 22(8)班 学号 3122008883

姓名 陈煜祺 协作者 无 教师评定

实验题目 四、第 7 章习题 15 题 (2) (3)

1、 题目内容

题目：

第7章习题 15题（2）（3）

考察点：

函数使用、数组、排序、折半查找

完成提示：

第（1）小题无需写函数，直接在main函数中定义以下数组即可。

职工信息：

```
char name[10][20] = {"John", "Mike", "Peter", "Judy", "Anny", "Lucy", "Ketty",  
"Davi", "Cook", "Diana"};
```

```
int num[10] = {3008, 3004, 3010, 3007, 3006, 3005, 3002, 3001, 3009, 3003};
```

第（2）小题使用选择排序。在main函数中输出姓名和职工号。

2、 题目分析与算法设计（分析思路、NS 图或伪代码）

本题要求实现的函数有“对已有数组进行选择排序”和“对已排序好的数组进行二分查找”两部分。对于每个函数，用伪代码表示的算法如下：

排序函数 void mysort(char name[][20], int num[]):

```
begin
    for i=0 to n-2:
        //记录当前最小的元素下标 k
        i⇒k
        for j=i to n-1:
            if num[j]<num[k]
                j⇒k
            end for
        交换 num[i] 和 num[k]
        交换 name[i] 和 name[k]
    end for
end
```

查找函数 char* myfind(int x, char name[][20], int num[], int l, int r):

```
begin
    if 已检索完毕: return 错误信息
    (l+r)/2⇒mid
    if num[mid]==x: //中间值为所查找值，直接输出
        return name[mid]
    if num[mid]>x: //中间值大于所查找值，在左半区间查找
        return myfind(x,name,num,l,mid)
    if num[mid]<x: //中间值小于所查找值，在右半区间查找
        return myfind(x,name,num,mid+1,r)
end
```

3、 代码实现

```
1  #include<stdio.h>
2  #include<string.h>
3
4  char err[20]="Not found.";
5
6  void mysort(char name[][20],int num[])
7  {
8      const int n=10;
9      for(int i=0;i<=n-2;i++)
10     {
11         int k=i;
12         for(int j=i;j<=n-1;j++)
13         {
14             if(num[j]<num[k])
15                 k=j;
16         }
17         int temp=num[k];
18         num[k]=num[i];
19         num[i]=temp;
20
21         char tmp[20];
22         strcpy(tmp,name[k]);
23         strcpy(name[k],name[i]);
24         strcpy(name[i],tmp);
25     }
26 }
27
28 char* myfind(int x,char name[][20],int num[],int l,int r)
29 {
30     if(l>r||r==0)
31         return err;
32     int mid=(l+r)/2;
33     if(num[mid]==x)
34         return name[mid];
35     if(x<num[mid])
36         return myfind(x,name,num,l,mid);
37     else
38         return myfind(x,name,num,mid+1,r);
39 }
40
41 int main()
42 {
43     char name[10][20] = {"John", "Mike", "Peter", "Judy", "Anny",
44                          "Lucy", "Ketty", "Davi", "Cook", "Diana"};
45     int num[10] = {3008, 3004, 3010, 3007, 3006,
46                   3005, 3002, 3001, 3009, 3003};
47
48     mysort(name,num);
49
50     int x;
51     scanf("%d",&x);
52     printf("%s",myfind(x,name,num,0,9));
53
54     return 0;
55 }
```


4、 实验结果（测试样例运行结果截图）

```
C:\Users\Administrator\Desktop\第四次.exe
3005
Lucy
Process returned 0 (0x0)    execution time : 2.321 s
Press any key to continue.

C:\Users\Administrator\Desktop\第四次.exe
3020
Not found.
Process returned 0 (0x0)    execution time : 8.270 s
Press any key to continue.
```

5、 问题与总结（实验中遇到的难以解决的问题、对实验题目考察知识点的总结）

本次实验题目将“选择排序”算法、“折半查找”算法和“以数组作为参数的函数”作为核心考察点，并附带考察了使用 `strcpy` 函数进行字符串交换等操作。实验要求我们熟练掌握函数的编写，并着重考察了以数组名作为函数参数进行传递的能力。

顺利完成本次实验题目的关键点在于两个算法的熟练运用和函数参数的正确传递。要顺利解决这个问题，需要我们理解选择排序算法的思想，并根据本次实验题目的要求，对课本中提供的选择排序原型进行微调；同时也要明确折半查找的边界条件和状态转移方式，正确地设立边界条件和新的查找区间，才能避免程序陷入死循环。

实验还渗透了用函数实现模块化程序设计的思想，对课程的学习大有裨益。

广东工业大学

先进制造学院 计算机科学与技术专业 22(8)班 学号 3122008883

姓名 陈煜祺 协作者 无 教师评定

实验题目 五、第 8 章习题 15 题

1、 题目内容

有一个班 4 个学生，5 门课程。

- ① 求第 1 门课程的平均分；
- ② 找出有两门以上课程不及格的学生，输出他们的学号和全部课程成绩及平均成绩；
- ③ 找出平均成绩在 90 分以上或全部课程成绩在 85 分以上的学生。

分别编写 3 个函数实现以上 3 个要求。

2、 题目分析与算法设计（分析思路、NS 图或伪代码）

对于各小题要求实现的函数，用伪代码表示的算法如下：

第 1 小题 `float avg1(int (*a)[5])`

```
begin
|   0⇒avg
|   for i=0 to 3:
|       |   avg+=a[i]
|   end for
|   return  avg/4
end
```

第 2 小题 `void findfail(int *a, int index)`

```
begin
|   0⇒cnt
|   for i=0 to 4:
|       |   if a[i]>60
|           |   cnt++
|       end for
|   if cnt>2:   输出该学生的学号(index)、各课程成绩、平均成绩
end
```

第 3 小题 `void findgood(int *a, int index)`

```
begin
|   0⇒cnt,1⇒flag
|   for i=0 to 4:
|       |   if a[i]>60:      cnt++
|       |   if a[i]<=85:    0⇒flag
|   end for
|   if cnt>2 or flag==1:   输出该学生的学号(index)、各课程成绩、平均成绩
end
```

3、代码实现

```
1  #include<stdio.h>
2
3  float avg1(int (*a)[5])
4  {
5      float avg=0;
6      for(int i=0;i<=3;i++)
7          avg+=a[i][0];
8      return avg/4;
9  }
10
11 void findfail(int *a, int index)
12 {
13     int cnt=0;
14     for(int i=0;i<=4;i++)
15         if(a[i]<60)
16             cnt++;
17     if(cnt>2)
18     {
19         float sum=0;
20         printf("第%d号学生, 五门课程成绩分别为: ", index);
21         for(int i=0;i<=3;i++)
22         {
23             sum+=a[i];
24             printf("%3d, ", a[i]);
25         }
26         sum+=a[4];
27         printf("%3d. ", a[4]);
28         printf("平均成绩为: %-5.2f\n", sum/5);
29     }
30 }
31
32 void findgood(int *a, int index)
33 {
34     int cnt=0, flag=1;
35     for(int i=0;i<=4;i++)
36     {
37         if(a[i]>90)
38             cnt++;
39         if(a[i]<=85)
40             flag=0;
41     }
42     if(cnt>2 || flag==1)
43     {
44         float sum=0;
45         printf("第%d号学生, 五门课程成绩分别为: ", index);
46         for(int i=0;i<=3;i++)
47         {
48             sum+=a[i];
49             printf("%3d, ", a[i]);
50         }
51         sum+=a[4];
52         printf("%3d. ", a[4]);
53         printf("平均成绩为: %-5.2f\n", sum/5);
54     }
55 }
56
57 int main()
58 {
59     int a[4][5]=
60     {
61         {0, 59, 58, 90, 100},
62         {86, 87, 88, 89, 90},
63         {23, 35, 47, 100, 100},
64         {99, 82, 91, 100, 88}
65     };
66
67     printf("第一门成绩的平均分是: %5.2f分\n\n", avg1(a));
68
69     printf("有两门以上课程成绩不及格的学生有: \n");
70     for(int i=0;i<=3;i++)
71         findfail(a[i], i+1);
72
73     printf("\n平均成绩在90分以上或全部课程在85分以上的学生有: \n");
74     for(int i=0;i<=3;i++)
75         findgood(a[i], i+1);
76
77     return 0;
78 }
```

4、 实验结果（测试样例运行结果截图）

```
C:\Users\Administrator\Desktop\第五次.exe
第一门成绩的平均分是：52.00分
有两门以上课程成绩不及格的学生有：
第1号学生，五门课程成绩分别为： 0, 59, 58, 90, 100。平均成绩为：61.40
第3号学生，五门课程成绩分别为： 23, 35, 47, 100, 100。平均成绩为：61.00
平均成绩在90分以上或全部课程在85分以上的学生有：
第2号学生，五门课程成绩分别为： 86, 87, 88, 89, 90。平均成绩为：88.00
第4号学生，五门课程成绩分别为： 99, 82, 91, 100, 88。平均成绩为：92.00
Process returned 0 (0x0)    execution time : 0.006 s
Press any key to continue.
```

5、 问题与总结（实验中遇到的难以解决的问题、对实验题目考察知识点的总结）

本次实验题目将难度中等，以函数的编写和数组的使用作为核心考察点，并附带考察了顺序输出数组各元素和求平均数等功能的实现。实验要求我们熟练掌握函数的编写，并着重考察了以数组名作为函数实参进行传递的能力。

顺利完成本次实验题目的关键点在于三个功能的正确实现。要顺利解决这个问题，需要我们打牢基本语法基础，并根据本次实验题目的要求，用基础的语法语句逐条搭建、组合出要实现的功能；同时也要明确数组名作为实参的传递方式，正确地设置形参，才能顺利地接收传入的实参，完成虚实结合。

本次实验充分体现了用函数实现模块化程序设计的思想，为将来更复杂的项目工程功能开发打下了初步的基础。

广东工业大学

先进制造学院 计算机科学与技术专业 22(8)班 学号 3122008883

姓名 陈煜祺 协作者 无 教师评定

实验题目 六、课本例 9.9

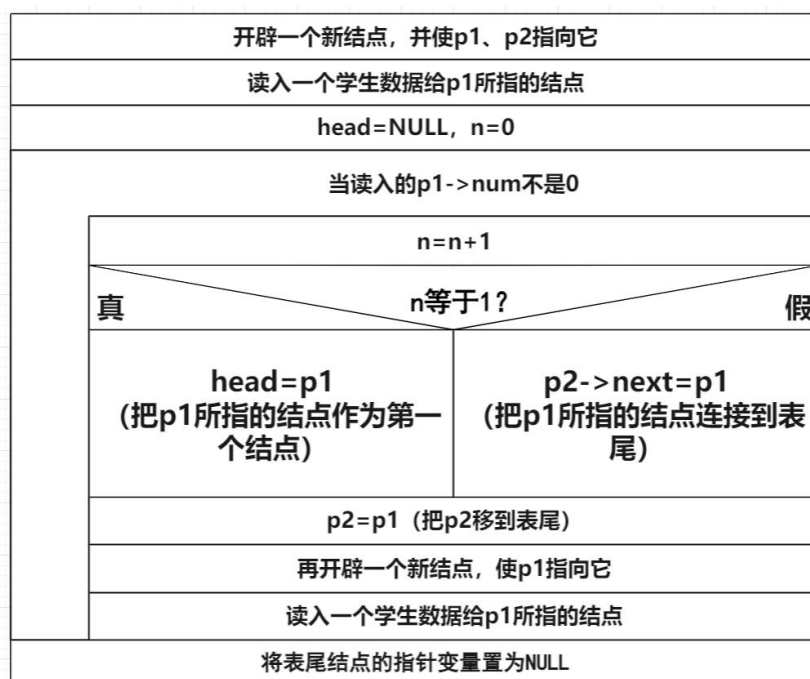
1、 题目内容

要求复现课本第 311 页，例题 9.9：

写一个函数建立一个有 3 名学生数据的单向动态链表。

2、 题目分析与算法设计（分析思路、NS 图或伪代码）

根据题目要求，编写 create()函数：建立有 n 个结点的单向动态链表，直到输入为 0 时停止生成新的结点、结束函数并返回链表的头指针。然后在 main 函数中遍历并输出 create()函数生成的有 n 个结点的链表。对于 create()函数，用 N-S 图表示的算法如下（依据图 9.10）：



在 `create()` 函数中，建立具有 3 个结点的单向动态链表时的流程细节，由下方的示意图展示（依据图 9.11 至图 9.14，但以测试样例数据为例）：

图 9.11

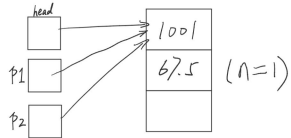


图 9.12

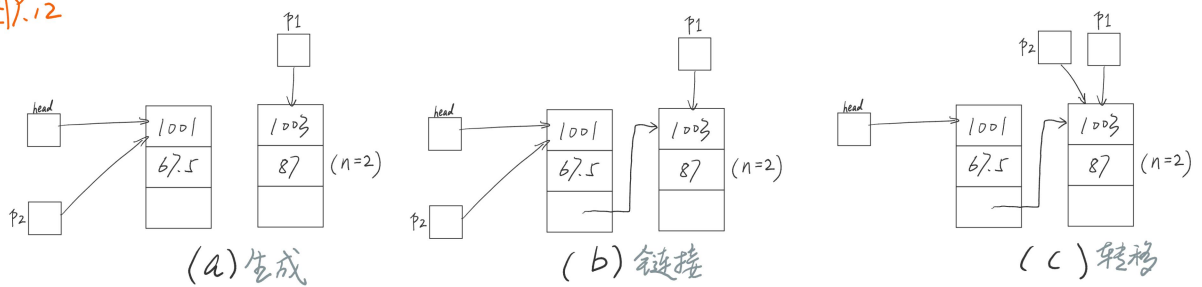


图 9.13

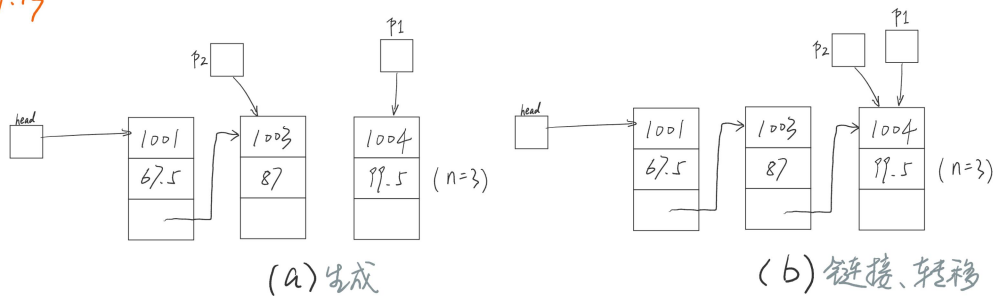
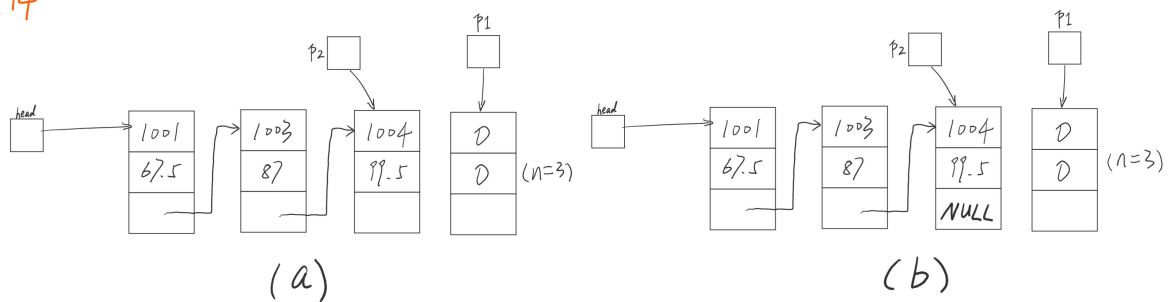


图 9.14



3、 代码实现

```
1  #include<stdio.h>
2  #include<stdlib.h>
3
4  struct Student{
5      long num;
6      float score;
7      struct Student* next;
8  };
9
10 int n;
11
12 struct Student* create()
13 {
14     struct Student *head=NULL,*p1,*p2;
15     n=0;
16
17     p1=p2=malloc(sizeof(struct Student));
18     scanf("%ld,%f",&p1->num,&p1->score);
19
20     while(p1->num!=0)
21     {
22         n++;
23         if(n==1)
24             head=p1;
25         else
26             p2->next=p1;
27         p2=p1;
28         p1=malloc(sizeof(struct Student));
29         scanf("%ld,%f",&p1->num,&p1->score);
30     }
31     free(p1);
32     p2->next=NULL;
33     return head;
34 }
35
36 int main()
37 {
38     struct Student* now=create();
39     while(now!=NULL)
40     {
41         printf("num:%5ld,score:%5.1f\n",now->num,now->score);
42         now=now->next;
43     }
44     return 0;
45 }
46
```


4、 实验结果（测试样例运行结果截图）

```
D:\学习\大一\程序设计实验课\第7次\第六题.exe
1001, 67.5
1003, 87
1004, 99.5
0, 0
num: 1001, score: 67.5
num: 1003, score: 87.0
num: 1004, score: 99.5

Process returned 0 (0x0)   execution time : 24.820 s
Press any key to continue.
```

5、 问题与总结（实验中遇到的难以解决的问题、对实验题目考察知识点的总结）

本次实验题目以单向动态链表的建立使用作为核心考察点，并综合考察了函数、结构体、指针、`malloc` 函数的使用等多个知识点，是对本学期程序设计课程所学知识和技能的一次综合考验。

顺利完成本次实验题目的关键点在于单向动态链表的合理实现。要解决这个问题，需要我们明确动态链表从无到有的建立过程中各指针的指向，从而合理地设计算法，并根据本次实验题目的要求，熟练运用已经掌握的函数、结构体、指针、`malloc/free` 函数等多基础语法将其实现。

本次实验首次尝试了动态链表的建立，是对数据结构的一次初步探索，为将来学习更加复杂的数据结构打下了牢靠的基础。