

第 3 章 分支结构

3.1 教学要点

本章通过典型程序解析，主要介绍分支结构程序设计的思想和实现方法，并介绍字符类型和逻辑运算等语言知识。使学生理解 `else-if` 语句和 `switch` 语句的执行机制以及能综合运用编写分支结构类的程序。

3.1 节通过引例“简单的猜数游戏”，主要介绍分支结构的特征以及实现分支结构的 `if-else` 语句和 `else-if` 语句三个知识点。教师在讲授时，应详细介绍上述三个知识点，特别是 `else-if` 语句的执行机制（流程），使学生能使用 `else-if` 结构进行多分支程序的设计。

3.2 节通过引例“四则运算”，引入字符类型、逻辑运算。教师在讲授时，应重点介绍字符类型的特征、字符的 ASCII 码、逻辑运算符以及逻辑表达式的表示方法，并结合示例说明对字符类型数据的处理方法。

3.3 节通过引例“查询自动售货机中商品的价格”，主要介绍 `switch` 语句以及多分支结构的程序设计。教师在讲授时，应重点介绍 `switch` 语句的语法结构，可根据 `break` 语句的使用方式分三种情况介绍，对 `switch` 后的表达式以及 `case` 后的常量表达式应作重点说明。使学生能使用 `switch` 结构进行分支程序的设计。在此基础上，分析多分支结构的程序设计方法，重点介绍嵌套的 `if` 结构实现方法，特别是 `else` 与 `if` 的配对问题。

讲授学时：4 学时，实验学时同讲授学时。

本章的知识能力结构图见图 3.1。

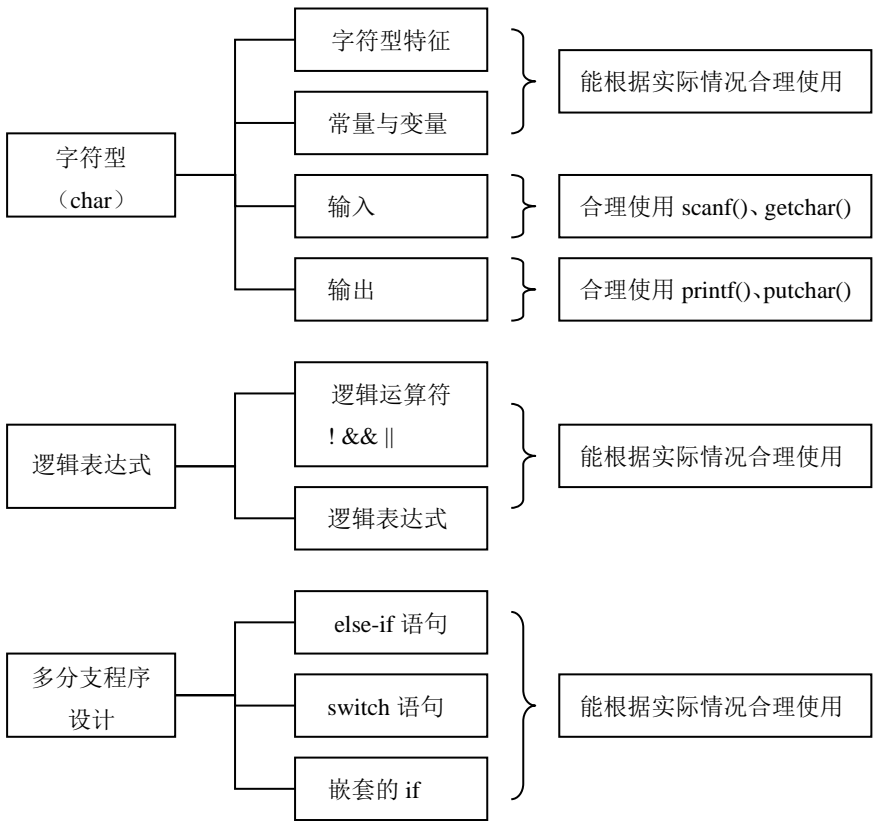






图 3.1 知识能力结构图

3.2 讲稿

1	 <h3>Chap 3 分支结构</h3> <p>3.1 简单的猜数游戏</p> <p>3.2 四则运算</p> <p>3.3 查询自动售货机中商品的价格</p>	本章分 3 节。
2	 <h3>本章要点</h3> <ul style="list-style-type: none"> ■ 什么是分支结构？它的作用是什么？ ■ switch 语句中的 break 起什么作用？ ■ 逻辑运算和关系运算的相同之处是什么？它们之间又有什么不同？ ■ 字符型数据在内存中是如何存储的？ 	提出本章的学习要点。
3	 <h3>3.1 简单的猜数游戏</h3> <div> <p>输入你所猜的整数（假定1~100内），与计算机产生的被猜数比较，若相等，显示猜中；若不等，显示与被猜数的大小关系</p> </div> <p>3.1.1 程序解析</p> <p>3.1.2 二分支结构和if – else语句</p> <p>3.1.3 多分支结构和else – if语句</p>	<p>借助引例提出要解决的问题，并简要分析，提供思路，指出与第 2.3 节学习的二分支结构的异同。</p> <p>可以通过运行例 3-1 程序，让学生感受多分支的场景。</p> <p>本节介绍为解决这个问题所编写的程序和涉及到的语言知识。</p>
4	 <h3>3.1.1 程序解析</h3> <p>例3-1 简单的猜数游戏</p> <p>输入你所猜的整数yournumber(假定1~100内)，与计算机产生的被猜数mynumber比较，若相等，显示猜中；若不等，显示与被猜数的大小关系。</p> <p>yournumber vs mynubmer: 3种情况：</p> <ul style="list-style-type: none"> ■ yournumber == mynumber if (==) Good Guess! ■ yournumber > mynumber else if (>) Too big! ■ yournumber < mynumber else Too small! 	<p>具体分析示例的处理流程，提出问题：</p> <p>猜数与被猜数的关系有几种情况？</p> <p>各种情况如何表述？</p> <p>如何一一对应结果？</p> <p>程序如何实现？</p>

5	<div style="display: flex; justify-content: space-between;"> <div> <pre>#include <stdio.h> int main (void) { int mynumber = 38; int yournumber; printf ("Input your number: "); scanf ("%d", &yournumber); if (yournumber == mynumber){ printf ("Good Guess!\n"); }else if (yournumber > mynumber){ printf ("Too big!\n"); }else{ printf ("Too small!\n"); } return 0; }</pre> </div> <div> <h3>源程序-猜数游戏</h3> <div style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> Input your number:48 Too big! </div> <div style="border: 1px solid black; padding: 5px;"> Input your number:38 Good Guess! </div> <div style="background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> 多层缩进的书写格式 使程序层次分明 </div> </div> </div>	<p>展示、运行例 3-1 程序，并概要解读程序：</p> <p>(1) 程序运行过程：</p> <p>输入一个整数；</p> <p>判断该整数与计算机指定数的关系，并显示结果。</p> <p>(2) 简要分析判断条件和 if 结构。</p> <p>(3) 提示：多层缩进的书写格式，使程序层次分明。</p> <p>提示：强烈推荐养成“{语句}”的好习惯，即使只有一条语句。如本例中 if 语句。在 Dev-C 环境下，这样写能自动调整对齐为多层缩进的格式，使程序层次分明。</p>
6	<div style="display: flex; align-items: center;"> <div style="flex: 1;"> <h3>3.1.2 二分支结构和 if-else 语句</h3> <div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="text-align: center;"> <p>if (表达式) 语句1</p> <p>else 语句2</p> <p>一条语句</p> </div> <div style="text-align: center;"> <p>if (表达式) 语句1</p> </div> </div> </div> <div style="flex: 1; border: 1px solid black; padding: 5px; margin-left: 10px;"> <p>真(非0) 假(0)</p> <p>表达式 表达式</p> <p>语句1 语句2</p> </div> </div>	<p>二分支结构说明：</p> <p>(1) 基本的 if 语句的一般形式；</p> <p>(2) 执行流程</p> <p>注意事项：</p> <p>语句 1 和语句 2 必须且只执行其中一个；</p> <p>else 部分可省略；</p> <p>if-else 是一条语句；</p> <p>语句 1 和语句 2 只允许一条语句，否则应用 {} 把这些语句括起来组成复合语句。</p> <p>提示：强烈推荐养成“{语句}”的好习惯，即使只有一条语句。</p>
7	<div style="display: flex; justify-content: space-between;"> <div> <h3>判断数字的奇偶性</h3> <p>例3-2奇偶分家</p> <p>输入一个正整数n，再输入n个非负整数，统计奇数和偶数各有多少个？</p> </div> <div> <div style="background-color: #e6f2ff; padding: 5px; margin-bottom: 10px;"> $number \% 2 == 0$ </div> <pre>count_odd = 0; count_even = 0; for(i = 1; i <= n; i++){ 读入一个非负整数number if (number能被2整除) /* 该数为偶数 */ count_odd++; else /* 该数为奇数 */ count_even++; }</pre> </div> </div>	<p>举例说明二分支结构程序设计。</p> <p>本例是标准的 if-else 结构。</p> <p>使用求余运算符“%”表达条件：</p> <p>如何判断一个数是偶数？</p> <p>$number \% 2 == 0$</p> <p>设问：</p> <p>如何判断一个数是奇数？</p> <p>$number \% 2 != 0$</p> <p>如何判断一个数是 3 的倍数？</p> <p>$number \% 3 == 0$</p> <p>.....</p>
8	<div style="display: flex; justify-content: space-between;"> <div> <pre>#include <stdio.h> int main (void) { int count_odd, count_even, i, n, number; count_odd = 0; count_even = 0; printf ("Enter n: "); scanf ("%d", &n); printf ("Enter %d numbers: ", n); for (i = 1; i <= n; i++){ scanf ("%d", &number); if (number % 2 != 0){ count_odd++; }else{ count_even++; } } printf ("Odd: %d, Even: %d \n", count_odd, count_even); return 0; }</pre> </div> <div> <h3>源程序-奇偶分家</h3> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> Enter n: 4 Enter 4 numbers: 5 8 101 9 Odd: 3, Even: 1 </div> </div> </div>	<p>展示、运行程序。</p> <p>提示：强烈推荐养成“{语句}”的好习惯，即使只有一条语句。如本例中 if 语句。在 Dev-C 环境下，这样写能自动调整对齐为多层缩进的格式，使程序层次分明。</p>

9	<p>统计学生的成绩</p> <p>例3-3 输入一个正整数n，再输入n个学生的成绩，计算平均分，并统计不及格成绩的个数。</p> <pre> for (i = 1; i <= n; i++){ 输入1个学生的成绩 score 累加成绩 total 统计不及格成绩的个数 count } 输出结果 </pre>	<p>本例综合运用循环和分支。建议先运行程序，使学生感受场景。</p> <p>首先引导学生理解问题的要求，输入什么？要求输出什么结果？如何处理？如何构建程序结构？抓住主要矛盾！</p> <p>为计算平均分需要计算 n 个学生的总成绩，故需要一个变量（total）来存放；为统计不及格人数，也需要一个变量（count）来计数。</p> <p>重复 n 次，每次输入 1 个成绩，然后累加其分值到 total，若不及格，则还需计数至 count。</p>
10	<p>源程序-统计成绩</p> <pre> #include <stdio.h> int main (void) { int count, i, n; double score, total; printf ("Enter n: "); scanf ("%d", &n); total = 0; count = 0; for (i = 1; i <= n; i++){ printf ("Enter score # %d: ", i); scanf ("%lf", &score); total = total + score; if (score < 60) { count++; } } if (n != 0) printf ("Average = %.2f\n", total / n); else printf ("Average = %.2f\n", 0.0); printf ("Number of failures = %d\n", count); return 0; } </pre> <p>Enter n: 4 Enter score #1: 67 Enter score #2: 54 Enter score #3: 88 Enter score #4: 73 Grade average = 70.50 Number of failures = 1</p> <p>此处省略else</p>	<p>展示、运行程序，注意提醒：</p> <p>（1）变量 total、count 初值置零；</p> <p>（2）for 循环体内 printf 函数是输入的提示信息；</p> <p>（3）最后运算结果必须在全部处理完后输出，故输出在 for 语句后。</p>
11	<p>3.1.3 多分支结构和else – if 语句</p> <p>else-if 语句是最常用的实现多分支（多路选择）的方法</p> <pre> if (表达式1) 语句1; else if (表达式2) 语句2; else if (表达式n-1) 语句n-1; else 语句n; </pre>	<p>重点说明：</p> <p>结合例 3-1，该例即有三个分支；else-if 语句是实现多路选择最常用的方法之一；</p> <p>详细解释 else-if 的一般形式。</p> <p>提示：else-if 语句的书写格式为左对齐。在 Dev-C 环境下，能自动调整对齐。</p>
12	<p>else – if 语句</p> <pre> if (表达式1) 语句1 else if (表达式2) 语句2 else if (表达式n-1) 语句n-1 else 语句n </pre> <p>n个分支需要n-1次比较</p>	<p>重点说明 else-if 语句的执行流程：</p> <p>按书写顺序逐个比较，一旦符合条件则执行相应语句。</p>

13	<p>更改例2-4中的分段计算水费的问题</p> <p>例3-4 例2-4中提出的分段计算水费的问题。居民应交水费y(元)与月用水量x(吨)的函数关系式修正如下,并编程实现。</p> $y = f(x) = \begin{cases} 0, & x < 0 \\ \frac{4x}{3}, & 0 \leq x \leq 15 \\ 2.5x - 10.5, & x > 15 \end{cases}$	<p>对例 2-4 分段计算水费问题的完善,增加输入小于 0 的情况。</p> <p>练习使用 else-if 语句实现。</p>
14	<p>源程序-分段计算水费</p> <pre>#include <stdio.h> int main (void) { double x, y; printf ("Enter x:"); scanf ("%lf", &x); if (x < 0){ y = 0; }else if (x <= 15){ y = 4 * x / 3; }else{ y = 2.5 * x - 10.5; } printf ("f(%.2f) = %.2f\n", x, y); return 0; }</pre> $y = f(x) = \begin{cases} 0 & x < 0 \\ \frac{4x}{3} & 0 \leq x \leq 15 \\ 2.5x - 10.5 & x > 15 \end{cases}$ <p>Enter x: -0.5 f(-0.50) = 0.00 Enter x: ?</p> <p>Enter x: 9.5 f(9.50) = 12.67 Enter x: ?</p> <p>Enter x: 21.3 f(21.30) = 42.75</p>	<p>分析程序结构、运行程序,输入 3 个不同区间值 x, 查看运行结果,必要时可单步运行程序。</p> <p>设问:</p> <p>是否还需要增加测试数据 x? 为什么? 若要增加,应输入多少?</p> <p>答: 最好再增加两组测试用例,因为尚未对分段函数参数的边界值进行测试。可再给出 $x=0$ 和 $x=15$ 时的两种情况。</p> <p>本例至少需要 5 组测试数据。</p>
15	<p>3.2 四则运算</p> <p>输入一个形式如“操作数 运算符 操作数”的四则运算表达式,输出运算结果。</p> <p>3.2.1 程序解析</p> <p>3.2.2 字符类型</p> <p>3.2.3 字符型数据的输入和输出</p> <p>3.2.4 逻辑运算</p>	<p>借助引例提出要解决的问题,并简要分析,提供思路,可与练习 2-9 整数四则运算进行比较,指出输入数据、运算结果的异同。</p> <p>可以通过运行例 3-5 程序,让学生感受场景。</p> <p>本节介绍为解决这个问题所编写的程序和涉及到的语言知识。</p>
16	<p>3.2.1 程序解析</p> <p>例3-5 输入一个形式如“操作数 运算符 操作数”的四则运算表达式,输出运算结果。要求对除数为0的情况作特别处理。</p> <p>输入: 3.1+4.8 输出: =7.90</p> <p>输入: 5.0-3.9 输出: =1.1</p> <pre>value1 op value2 (op: 存放一个字符 + - * / 等) if(op == '+') value1 + value2 else if (op == '-') value1 - value2 else if (op == '*') value1 * value2 else if (op == '/') value1 / value2 else "Unknown operator"</pre>	<p>具体分析示例的处理流程,提出问题:</p> <p>输入数据的特点?</p> <p>如何根据输入字符判断所对应的运算?</p> <p>程序如何实现?</p>

```

17
# include <stdio.h>
int main (void)
{ double value1, value2;
  char op;
  printf ("Type in an expression: ");
  scanf ("%lf%lc%lf", &value1, &op, &value2);
  if (op == '+')
    printf ("=%.2fn", value1 + value2);
  else if (op == '-')
    printf ("=%.2fn", value1 - value2);
  else if (op == '*')
    printf ("=%.2fn", value1 * value2);
  else if (op == '/')
  {
    if (value2 != 0)
      printf ("=%.2fn", value1 / value2);
    }
    else{
      printf ("Divisor can not be 0!");
    }
  }
  printf ("Unknown operator!\n");
}
return 0;
}

```

源程序-四则运算

Type in an expression: 3.1+4.8
=7.90

Type in an expression: 3.4/0
Divisor can not be 0!

展示、运行例 3-5 程序，并概要解读程序：

(1) 程序运行过程:

根据输入的运算符判断执行相应的运算，并输出结果。

(3) 采用 else-if 语句实现。

```


18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043

```

由例 3-5 程序引出字符型数据。

(1) 字符类型 `char` 和之前学习的整型 `int`、单精度浮点型 `float`、双精度浮点型 `double` 一样，是 C 语言基本数据类型之一。

char op; 定义 op 为 char (字符) 型变量

19																																																																																																																																										
	'a' 'z' 'A' 'Z' '0' '9' ' ' '\n'	字符常量																																																																																																																																								
	ASCII字符集：列出所有可用的字符																																																																																																																																									
	每个字符：惟一的次序值（ASCII 码）																																																																																																																																									
		ASCII 码表																																																																																																																																								
	<table border="1"> <thead> <tr> <th>符号</th><th>10进制</th><th>符号</th><th>10进制</th><th>符号</th><th>10进制</th><th>符号</th><th>10进制</th></tr> </thead> <tbody> <tr><td>'0'</td><td>48</td><td>P</td><td>80</td><td>x</td><td>120</td><td>~</td><td>126</td></tr> <tr><td>'1'</td><td>49</td><td>Q</td><td>81</td><td>a</td><td>97</td><td>z</td><td>122</td></tr> <tr><td>'A'</td><td>65</td><td>R</td><td>82</td><td>b</td><td>98</td><td>x</td><td>123</td></tr> <tr><td>'Z'</td><td>90</td><td>S</td><td>83</td><td>c</td><td>99</td><td>x</td><td>124</td></tr> <tr><td>' '</td><td>32</td><td>T</td><td>84</td><td>d</td><td>100</td><td>y</td><td>125</td></tr> <tr><td>'a'</td><td>97</td><td>U</td><td>85</td><td>e</td><td>101</td><td></td><td>117</td></tr> <tr><td>'z'</td><td>122</td><td>V</td><td>86</td><td>f</td><td>102</td><td>v</td><td>116</td></tr> <tr><td></td><td></td><td>W</td><td>87</td><td>g</td><td>103</td><td>w</td><td>115</td></tr> <tr><td></td><td></td><td>X</td><td>88</td><td>h</td><td>104</td><td>x</td><td>120</td></tr> <tr><td></td><td></td><td>Y</td><td>89</td><td>i</td><td>105</td><td>y</td><td>121</td></tr> <tr><td></td><td></td><td>Z</td><td>90</td><td>j</td><td>106</td><td>z</td><td>122</td></tr> <tr><td></td><td></td><td>[</td><td>91</td><td>k</td><td>107</td><td>[</td><td>123</td></tr> <tr><td></td><td></td><td>\</td><td>92</td><td>l</td><td>108</td><td>\</td><td>124</td></tr> <tr><td></td><td></td><td>]</td><td>93</td><td>m</td><td>109</td><td>]</td><td>125</td></tr> <tr><td></td><td></td><td>-</td><td>94</td><td>n</td><td>110</td><td>-</td><td>126</td></tr> <tr><td></td><td></td><td>_</td><td>95</td><td>o</td><td>111</td><td>_</td><td>127</td></tr> </tbody> </table>	符号	10进制	符号	10进制	符号	10进制	符号	10进制	'0'	48	P	80	x	120	~	126	'1'	49	Q	81	a	97	z	122	'A'	65	R	82	b	98	x	123	'Z'	90	S	83	c	99	x	124	' '	32	T	84	d	100	y	125	'a'	97	U	85	e	101		117	'z'	122	V	86	f	102	v	116			W	87	g	103	w	115			X	88	h	104	x	120			Y	89	i	105	y	121			Z	90	j	106	z	122			[91	k	107	[123			\	92	l	108	\	124]	93	m	109]	125			-	94	n	110	-	126			_	95	o	111	_	127	
符号	10进制	符号	10进制	符号	10进制	符号	10进制																																																																																																																																			
'0'	48	P	80	x	120	~	126																																																																																																																																			
'1'	49	Q	81	a	97	z	122																																																																																																																																			
'A'	65	R	82	b	98	x	123																																																																																																																																			
'Z'	90	S	83	c	99	x	124																																																																																																																																			
' '	32	T	84	d	100	y	125																																																																																																																																			
'a'	97	U	85	e	101		117																																																																																																																																			
'z'	122	V	86	f	102	v	116																																																																																																																																			
		W	87	g	103	w	115																																																																																																																																			
		X	88	h	104	x	120																																																																																																																																			
		Y	89	i	105	y	121																																																																																																																																			
		Z	90	j	106	z	122																																																																																																																																			
		[91	k	107	[123																																																																																																																																			
		\	92	l	108	\	124																																																																																																																																			
]	93	m	109]	125																																																																																																																																			
		-	94	n	110	-	126																																																																																																																																			
		_	95	o	111	_	127																																																																																																																																			

字符型常量的特征：用一对单引号括起来，如'a'表示字母（字符）a。

ASCII 字符集列出了所有可用的字符：**ASCII** 码是每个字符唯一的次序值，由此可知比较字符的大小就是比较 **ASCII** 码的大小。

提示 2: 区分数值 1 和数字字符'1'。

20

字符变量

```
char op;  
定义字符型变量op，用于存放字符型数据  
op = '+';  
  
char op1, op2;  
op1 = 'A';  
op2 = op1;
```

重点说明变量 op 的特征：char（字符）类型、1 个字节、只能存放 1 个字符等。

21	<h3>3.2.3 字符型数据的输入和输出</h3> <ul style="list-style-type: none"> 调用scanf和printf输入输出字符 <pre>double value1, value2; char operator; printf("Type in an expression: "); scanf("%lf%c%lf", &value1, &op, &value2); printf("%.2f %c %.2f", value1, op, value2);</pre> <div> Type in an expression: 10.0+5.61 10.00 + 5.61 </div>	<p>scanf()函数和 printf()函数也可以处理字符型数据的输入和输出，与整型数据的输入与输出相比，说明两者的区别（介绍格式控制符%c）。</p> <p>设问： 输入表达式 10.0+5.61 时，在操作数和运算符间不能出现空格，为什么？</p> <p>解答：空格本身也是一个字符，如果输入空格，会被作为输入字符，即 op 的值为空格(' '), 而不是'+'。</p>
22	<ul style="list-style-type: none"> 字符输入函数getchar () 输入一个字符 字符输出函数putchar () 输出一个字符 <pre>char ch; ch = getchar (); putchar (ch); putchar ('?');</pre> <div> a a? </div> <p>只能处理单个字符的输入输出 即调用一次函数，只能输入或者输出一个字符</p>	<p>重点说明： getchar()和 putchar()函数的功能； getchar()和 putchar()函数的一般调用格式； 提醒：</p> <ol style="list-style-type: none"> （1）输入和输出时没有字符两侧的单引号； （2）getchar()和 putchar()函数只能处理单个字符的输入和输出。
23	<h3>输出一批字符</h3> <p>输入8个字符，然后将这些字符输出，输出时在字符之间加一个减号，第一个字符的前面和最后一个字符的后面都没有减号。</p> <pre>for (k = 1; k <= 8; k++){ ch = getchar (); putchar (ch); putchar ('-'); }</pre> <pre>for (k = 1; k <= 8; k++){ ch = getchar (); if (ch 是第一个字符) putchar (ch); else{ putchar ('-'); putchar (ch); } }</pre> <div> AMEHYST A-M-E-T-H-Y-S-T </div>	<p>输出格式示例：对首个输出项单独判断处理</p>
24	<h3>源程序段-输出一批字符</h3> <pre>char ch; int first = 1, k; /* first为1表示将要处理第1个字符 */ printf("Enter 8 characters: "); for(k = 1; k <= 8; k++){ ch = getchar(); if (first == 1){ putchar(ch); first = 0; /* first为0表示将要处理第2个及其后字符 */ }else{ putchar('-'); putchar(ch); } }</pre> <div> Enter 8 characters: AMETHYST A-M-E-T-H-Y-S-T </div>	<p>详细实现代码，使用标志变量 first</p>

25

3.2.4 逻辑运算

-1 <= x <= 1

0

-1

1

x

x >= -1 并且 x <= 1

x >= -1 && x <= 1

判断英文字母:

char ch;

printf("Enter a character: ");

ch = getchar();

if ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z')){

printf("It is a letter.\n");

}else{

printf("It is not a letter.\n");

}

Enter a character: d

It is a letter.

Enter a character: ?

It is not a letter.

由关系运算（比较 2 个值）引入逻辑运算（比较多个值）：

（1）数学式： $-1 \leq x \leq 1$ ，比较 3 个值 -1、x、1，相应的 C 表达式使用逻辑运算符与（&&）；

（2）判断键盘输入的字符 ch 是否为英文字母？

由于英文字符分大小写，故 ch 是小写字母或者 ch 是大写字母两种情况都属于 ch 是英文字母，进而引入逻辑运算符或（||）。

26

3种逻辑运算符

逻辑与 &&

逻辑或 ||

逻辑非 !

X && Y

X

Y

X || Y

X

Y

! X

X

结合数学中的集合概念，介绍 3 种逻辑运算符的含义与功能。

27

逻辑运算符的含义

逻辑与 &&

逻辑或 ||

逻辑非 !

$(x > 1) \&\& (y > 1)$

$(x > 1) || (y > 1)$

$(x > 1)$

29	<div data-bbox="304 203 491 277" data-label="Section-Header"> <h2>逻辑表达式</h2> </div> <p>逻辑表达式： 用逻辑运算符将逻辑运算对象连接起来的式子。</p> <pre>(ch >= 'a') && (ch <= 'z')</pre> <p>判断ch 是否为小写英文字母</p> <p>或：</p> <pre>ch >= 'a' && ch <= 'z'</pre> <pre>(ch >= 'a' && ch <= 'z') (ch >= 'A' && ch <= 'Z')</pre> <p>判断ch 是否为英文字母，分大小写</p>	<p>逻辑表达式与关系表达式一样，需要能够在程序中合理的使用，即正确描述条件。</p> <p>示例 1: 描述 ch 是小写字母的逻辑表达式； 示例 2: 描述 ch 是字母的逻辑表达式。 注意： 逻辑运算符的优先级低于关系运算符。</p>
30	<div data-bbox="304 620 501 692" data-label="Section-Header"> <h2>条件的表示</h2> </div> <p>例3-6 写出满足下列条件的C表达式。</p> <ul style="list-style-type: none"> ■ ch 是空格或者回车 <code>ch == ' ' ch == '\n'</code> ■ ch 是数字字符 <code>ch >= '0' && ch <= '9'</code> ■ year 是闰年，即 year 能被 4 整除但不能被 100 整除，或 year 能被 400 整除 <code>(year%4 == 0 && year%100 != 0) (year%400 == 0)</code> 	<p>举例练习描述各种条件的 C 语言表达式，包括关系表达式和逻辑表达式。</p>
31	<div data-bbox="304 1037 523 1108" data-label="Section-Header"> <h2>分类统计字符</h2> </div> <p>例3-7 输入n个字符，统计其中英文字母、数字字符和其他字符的个数。</p> <pre>for (i = 1; i <= n; i++){ 输入1个字符 ch if (ch是英文字母) letter++; else if(ch是数字字符) digit++; else other++; }</pre> <p>输出分类统计结果</p>	<p>本例综合运用循环和分支。建议先运行程序，使学生感受场景。</p> <p>可与例 3-3 对比分析。</p> <p>分类统计字符：</p> <p>（1）重复 n 次，每次输入 1 个字符，处理一个字符。</p> <p>（2）处理字符时：用 else-if 语句判断字符的类别，再分别计数。统计英文字母的个数，用变量 letter 计数；统计数字字符的个数，用变量 digit 计数；统计其他字符的个数，用变量 other 计数。</p> <p>（3）循环结束后，输出结果。</p>
32	<div data-bbox="304 1534 810 1915" data-label="Complex-Block"> <div data-bbox="304 1534 526 1915" data-label="Text"> <pre>#include <stdio.h> int main (void) { int digit, i, letter, n, other; char ch; digit = letter = other = 0; printf ("Enter n:"); scanf ("%d", &n); getchar (); printf ("Enter %d characters:", n); for (i = 1; i <= n; i++){ ch = getchar(); if ((ch >= 'a' && ch <= 'z') (ch >= 'A' && ch <= 'Z')) letter ++; else if (ch >= '0' && ch <= '9') digit ++; else other ++; } printf ("letter=%d,digit=%d,other=%d\n", letter, digit, other); return 0; }</pre> </div> <div data-bbox="542 1534 810 1915" data-label="Text"> <p>源程序-统计字符</p> <pre>Enter n: 10 Enter 10 characters: Reold 12-3 letter=5, digit=3, other=2</pre> <p><i>/* 读入并舍弃换行符 */</i></p> </div> </div>	<p>展示、运行程序，注意提醒：</p> <p>（1）变量 letter、digit、other 初值置零； （2）字符的输入方式； （3）用 else-if 语句分类计数； （4）表示类别的逻辑表达式。</p>

33	<p>3.3 查询自动售货机中商品的价格</p> <p>例3-8 查询自动售货机中商品的价格</p> <p>3.3.1 程序解析</p> <p>3.3.2 switch语句</p> <p>3.3.3 多分支结构</p>	<p>本节以“查询自动售货机中商品的价格”为例，介绍 switch 语句以及多分支结构的程序设计方法。</p>
34	<p>3.3.1 程序解析</p> <p>例3-8假设自动售货机出售4种商品，薯片(crisps)、爆米花(popcorn)、巧克力、可乐。显示菜单 售价分别是每份3.0、2.5、4.0、3.5。 在屏幕上显示菜单 [1] Select crisps [2] Select popcorn [3] Select chocolate [4] Select cola [0] Exit 用户可以连续查询商品的价格，直到输入0退出查询；不到5次时，用户可以选择退出。 当用户输入编号1~4，显示相应商品的价格；输入0，退出查询；输入其他编号，显示价格为0。</p> <pre> for (i = 1; i <= 5; i++){ 输入选项choice if (choice == 0) 退出循环 若 choice == 1~4 → price赋单价 == 其他 → price赋0 输出单价 } </pre>	<p>本例综合运用循环和分支。建议先运行程序，使学生感受场景。</p> <p>(1) 重复 5 次，每次先显示菜单，再输入选项 choice，并根据选项 choice 的值选择处理路径。</p> <p>(2) 处理时： 若 choice 的值为零，则提前结束循环； 否则，根据 choice 的值分别对 price 赋值单价或零，并输出。</p> <p>练习用 else-if 实现。</p>
35	<pre> #include <stdio.h> int main(void) { int choice, i; double price; printf("[1] Select crisps\n"); printf("[2] Select popcorn\n"); printf("[3] Select chocolate\n"); printf("[4] Select cola\n"); printf("[0] exit\n"); for(i = 1; i <= 5; i++){ printf("Enter choice:"); scanf("%d", &choice); if(choice == 0) break; switch(choice){ case 1: price = 3.0; break; case 2: price = 2.5; break; case 3: price = 4.0; break; case 4: price = 3.5; break; default: price = 0.0; break; } printf("price = %.1f\n", price); } printf("Thanks\n"); } </pre> <p>显示菜单 for (i = 1; i <= 5; i++){ 输入选项choice if (choice == 0) 退出循环 若 choice == 1~4 → price赋单价 == 其他 → price赋0 输出单价 } [1] Select crisps [2] Select popcorn [3] Select chocolate [4] Select cola [0] Exit Enter choice: 1 price = 3.0 Enter choice: 7 price = 0.0 Enter choice: 0 Thanks</p>	<p>展示、运行程序。</p> <p>简要说明如何使用 switch 语句，根据选项 choice 的值，确定变量 price 的值。从而引出多路选择的另一种实现方法。</p>
36	<p>3.3.2 switch语句</p> <p>处理多分支选择问题，3种情况</p> <p>1.在switch语句的每个语句段中都使用break语句</p> <pre> switch(表达式){ case 常量表达式1: 语句段1; break; case 常量表达式2: 语句段2; break; case 常量表达式n: 语句段n; break; default: 语句段n+1; break; } </pre>	<p>说明 switch 语句的功能、一般形式。点出 break 语句是 switch 语句中的关键，并根据 break 的不同使用方式，形成 3 种不同的情况。</p> <p>首先介绍各个 case 语句段都使用 break 的情况。</p> <p>注意: switch 后的表达式是一个测试对象; case 后是一个常量表达式，此处需重点说明何为常量表达式，后跟冒号 (:); break 的作用; default 的作用。</p>

<p>37</p>	<div> <div> <p>switch(表达式){</p> <p> 常量表达式的值不重复</p> <p> case 常量表达式1: 语句段1; break;</p> <p> case 常量表达式2: 语句段2; break;</p> <p> </p> <p> case 常量表达式n: 语句段n; break;</p> <p> default: 语句段n+1; break;</p> <p>}</p> </div> <div> <p>switch(choice){</p> <p> case 1: price = 3.0; break;</p> <p> case 2: price = 2.5; break;</p> <p> case 3: price = 4.0; break;</p> <p> case 4: price = 3.5; break;</p> <p> default: price = 0.0; break;</p> <p>}</p> </div> <div> <p>用else-if如何实现?</p> </div> </div>	<p>结合例 3-8，详细解释带有 break 语句的 switch 语句执行机制（流程图）；使学生理解 switch 后的测试表达式与 case 后的常量表达式的判断方式以及 break 语句的作用。</p> <p>设问：</p> <p>用 else-if 语句如何实现例 3-8 中的 switch 语句段？分析两者的异同？</p>
<p>38</p>	<div> <div> <p>两个数的简单计算器</p> <p>例3-9 编写一个简单计算器程序，可根据输入的运算符，对两个整数进行加、减、乘、除和求余运算，请对除数为0的情况作特别处理。要求使用 switch 语句编写。</p> <p>输入：-7/2 输出：-3</p> </div> <div> <pre> value1 op value2 (op: 存放一个字符 + - * / %等) switch (op){ case '+': value1 + value2 case '-': value1 - value2 case '*': value1 * value2 case '/': value1 / value2 case '/': value1 / value2 default: "Unknown operator" } </pre> </div> </div>	<p>例 3-9 两个数的简单计算器。</p> <p>（1）引导学生想象程序运行后输入什么？输出什么？</p> <p>（2）如何用 switch 语句来实现？switch 后的测试表达式是什么？</p> <p>（3）除数为 0 如何处理？</p>
<p>39</p>	<div> <div> <pre> #include <stdio.h> int main(void) { int value1, value2; char op; printf ("Type in an expression: "); scanf ("%d%c%d", &value1, &op, &value2); switch (op){ case '+': printf ("%d\n", value1 + value2); break; case '-': printf ("%d\n", value1 - value2); break; case '*': printf ("%d\n", value1 * value2); break; case '/': if (value2 != 0) printf ("%d\n", value1 / value2); else printf ("Divisor can not be 0\n"); break; case '%': if (value2 != 0) printf ("%d\n", value1 % value2); else printf ("Divisor can not be 0\n"); break; default: printf ("Unknown operator\n"); break; } return 0; } </pre> </div> <div> <p>源程序</p> <p>Type in an expression: -7/2 =-3</p> <p>错误: case op == '+'</p> </div> </div>	<p>展示、运行程序，重点介绍 switch 语句段。</p>
<p>40</p>	<div> <div> <p>2. 在switch中不使用break</p> <pre> switch(表达式){ case 常量表达式1: 语句段1; case 常量表达式2: 语句段2; ... case 常量表达式n: 语句段n; default: 语句段n+1; } </pre> </div> </div>	<p>介绍 switch 语句的第 2 种形式：不使用 break 语句。</p> <p>从形式上看，此种情形与前一种的区别在于缺少了 break 语句，可知 switch 语句中 break 语句可省略。</p>

41	<div> <pre>switch (表达式){ case 常量表达式1: 语句段1; case 常量表达式2: 语句段2; case 常量表达式n: 语句段n; default : 语句段n+1; }</pre> <pre>switch (choice) { case 1: price = 3.0; case 2: price = 2.5; case 3: price = 4.0; case 4: price = 3.5; default: price = 0.0; }</pre> </div>	<p>比较前一种形式的流程图，无 break 语句时，执行流程没有跳出 switch 语句，而是继续执行下一个 case 语句段。</p> <p>设问：</p> <p>设 choice 值为 2，则执行该形式 switch 后，price 值为多少？为什么？</p> <p>由此可见，若要终止 switch 语句的继续执行，可使用 break 语句；否则将继续执行其后的语句段。</p>
42	<h3>3. 在switch的某些语句段中使用break</h3> <p>例3-10 输入n个字符，分别统计出其中空格或回车、数字字符和其他字符的个数。</p> <p>比较：例3-7 输入n个字符，统计其中英文字母、数字字符和其他字符的个数。</p>	<p>比较前两种的 switch 形式可知：break 可省略，若不省略则可终止 switch 语句执行。这就产生了第三种形式：灵活使用 break，这也是最常用的一种形式。</p> <p>以例 3-10 为例说明，并和例 3-7 进行比较。</p>
43	<div> <pre>#include <stdio.h> int main(void) { int blank, digit, i, n, other; char ch; blank = digit = other = 0; printf("Enter n:"); scanf("%d", &n); getchar(); /* 读入并舍弃换行符 */ printf("Enter %d characters:", n); for (i = 1; i <= n; i++){ ch = getchar(); switch (ch){ case '\n': case '\n': blank++; break; case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': digit++; break; default: other++; break; } } printf("blank=%d, digit=%d, other=%d\n", blank, digit, other); return 0; }</pre> <p style="text-align: right;">源程序</p> <div> Enter n: 15 Enter 15 characters: Reold 12 or 45T blank=3, digit=4, other=8 </div> </div>	<p>说明：</p> <p>(1) case 常量表达式：后语句可以为空，共用后面的语句段，直到碰到 break 语句为止；</p> <p>(2) 比较例 3-7，本例没要求统计英文字母的数量。因为若要统计，则必须一一列出所有 52 个大小写英文字母，过于繁琐。由此可见，应根据具体问题选择适合的多分支实现方式，switch 或者 else-if。</p>
44	<h3>3.3.3 多分支结构</h3> <ul style="list-style-type: none"> 分支结构一般分为二分支和多分支两种结构 二分支结构用基本的 if 语句实现 多分支结构用实现方法： <ul style="list-style-type: none"> else - if 语句 switch语句 嵌套的 if - else语句 	<p>本节综合介绍分支结构程序的设计。</p> <p>分支结构可简单分为二分支和多分支两种结构。</p> <p>对于二分支结构，使用基本的 if-else 语句；对于多分支结构，除用前面讲的 else-if 语句和 switch 语句外，还可以用嵌套的 if 语句实现。</p>

45	<p>嵌套的 if - else 语句</p> <pre> if (表达式) 语句1 else 语句2 </pre> <p>if 语句 if 语句</p> <pre> if (表达式1) if (表达式2) 语句1 else 语句2 else if (表达式3) 语句3 else 语句4 </pre>	<p>当基本的 if-else 语句中的语句 1 或语句 2 是另一条基本的 if 语句时，就构成了嵌套的 if-else 结构。</p> <p>示例是 4 分支结构。</p> <p>提醒：</p> <p>若语句 1~语句 4 仍是基本的 if 语句，可以实现更多分支的选择结构。</p>
46	<p>else 和 if 的匹配</p> <pre> if (表达式1) if (表达式2) 语句1 else 语句2 else if (表达式3) 语句3 else 语句4 </pre> <p>else 与最靠近它的、没有与别的 else 匹配过的 if 匹配</p> <pre> if (表达式1) if (表达式2) 语句1 else if (表达式3) 语句3 else 语句4 </pre>	<p>先观察一般形式下嵌套的 if 结构，查看 else 与 if 的匹配情况。</p> <p>当省略从上往下第一条 else 语句时，else 与 if 是如何匹配呢？</p> <p>重点强调匹配准则：else 与最靠近它的、没有与别的 else 匹配过的 if 匹配。</p> <p>故省略第一条 else 语句时，程序段等价于 ppt 下方程序段，显然程序功能完全改变。</p>
47	<p>改变 else 和 if 的配对</p> <p>例3-11 改写下列 if 语句，使 else 和第1个 if 配对。</p> <pre> if (x < 2) if (x < 1) y = x + 1; else y = x + 2; </pre> <p>每条语句的执行条件？</p> <div style="display: flex; justify-content: space-around;"> <pre> if (x < 2){ if (x < 1) y = x + 1; } else y = x + 2; </pre> <pre> if (x < 2) if (x < 1) y = x + 1; else; else y = x + 2; </pre> </div>	<p>若要改变 else 正常的配对规则，一般采用两种方法：</p> <p>采用 {}，构造一个复合语句；</p> <p>增加空的 else 语句。</p>
48	<p>本章总结</p> <ul style="list-style-type: none"> 理解 if 语句和 switch 语句的执行机制 能使用关系表达式、逻辑表达式描述条件 能编写分支结构程序 <ul style="list-style-type: none"> 分支结构 <ul style="list-style-type: none"> 二分支：if-else 语句 多分支：else if、嵌套的 if-else、switch switch 语句 <ul style="list-style-type: none"> case 后为常量表达式，其值不重复 break 的使用 数据类型：char 型 运算符与表达式 <ul style="list-style-type: none"> 逻辑运算符、逻辑表达式 综合程序设计（分支结构） 	<p>归纳总结本章的主要知识点。</p>

3.3 练习与习题参考答案

3.3.1 练习参考答案

3-1 例 3-4 中使用 else-if 语句求解多分段函数，为了检查 else-if 语句的三个分支是否正确，已经设计了三组测试用例，请问还需要增加测试用例吗？为什么？如果要增加，请给出具体的测试用例并运行程序。

解答：最好再增加两组测试用例，因为尚未对分段函数参数的边界值进行测试。可再给出 $x=0$ 和 $x=15$ 时的两种情况。

3-2 计算符号函数的值：输入一个整数 x ，计算并输出符号函数 $\text{sign}(x)$ 的值。试编写相应程序

$$y = \text{sign}(x) = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases}$$

解答：

```
#include<stdio.h>
int main (void)
{
    int x, y;

    scanf ("%d",&x);
    if (x > 0) y = 1;
    else if (x == 0) y = 0;
    else y = -1;
    printf ("sign(%d) = %d\n",x,y);

    return 0;
}
```

3-3 统计学生平均成绩与及格人数：输入一个正整数 n ，再输入 n 个学生的成绩，计算平均成绩，并统计所有及格学生的人数。试编写相应程序。

解答：

```
#include <stdio.h>
int main (void)
{
    int count, grade, i, n;
    double average, sum;

    scanf ("%d", &n);
    count = 0;
    sum = 0;
```

```

    for (i = 1; i <= n; i++){
        scanf ("%d", &grade);
        sum = sum + grade;
        if (grade >= 60) {
            count++;
        }
    }
    if (n > 0) {
        average = sum / n;
    }
    else{
        average = 0;
    }
    printf ("average = %.1f\n", average);
    printf ("count = %d\n", count);

    return 0;
}

```

3-4 统计字符：输入 10 个字符，统计其中英文字母、空格或回车、数字字符和其他字符的个数。试编写相应程序。

解答：

```

#include<stdio.h>
int main (void)
{
    int blank, digit, i, letter, other;
    char ch;

    blank = digit = letter = other = 0;
    for (i = 1; i <= 10; i++){
        ch = getchar();
        if (ch >= 'a' && ch <= 'z' || ch >= 'A' && ch <= 'Z'){
            letter++;
        }
        else if (ch >= '0' && ch <= '9'){
            digit++;
        }
        else if (ch == ' ' || ch == '\n'){
            blank++;
        }
        else{
            other++;
        }
    }
}

```



```

printf ("letter = %d, blank = %d, digit = %d, other = %d\n", letter, blank, digit, other);

return 0;
}

```

3-5 输出闰年：输出 21 世纪中截止某个年份之前的所有闰年年份。判断闰年的条件是：能被 4 整除但不能被 100 整除，或者能被 400 整除。试编写相应程序。

解答：

```

#include <stdio.h>
int main (void)
{
    int flag, i, leap, year;

    scanf ("%d", &year);
    flag = 0;
    if (year <= 2000 || year > 2100){
        printf ("Invalid year!\n");
    }
    else{
        for (i = 2001; i <= year; i++) {
            if ((i % 4 == 0 && i % 100 != 0) || i % 400 == 0) {
                flag = 1;
                printf ("%d\n", i);
            }
        }
        if (flag == 0){
            printf ("None\n");
        }
    }

    return 0;
}

```

3-6 在例 3-8 程序中，如果把 switch 语句中所有的 break 都去掉，运行结果会改变吗？如果有变化，输出什么？为什么？

解答：如果去掉所有的 break 语句，运行结果会改变，输出 price = 0.0，因为不管 choice 值与其中某个常量表达式相等，当去掉 break 语句时，其后的所有语句段都将运行，故每次都将执行到 price=0.0 语句为止。

3-7 成绩转换：输入一个百分制成绩，将其转换为五分制成绩。百分制成绩到五分制成绩的转换规则：大于或等于 90 分为 A，小于 90 分且大于或等于 80 分为 B，小于 80 分且大于或等于 70 为 C，小于 70 分且大于或等于 60 为 D，小于 60 分为 E。试编写相应程序。

解答：

```

#include <stdio.h>

```

```

int main (void)
{
    int grade;
    char result;

    scanf ("%d", &grade);
    switch (grade/10){
        case 10:
            case 9: result = 'A'; break;
            case 8: result = 'B'; break;
            case 7: result = 'C'; break;
            case 6: result = 'D'; break;
            default: result = 'E';
        }
    printf ("%c\n", result);

    return 0;
}

```

3-8 查询水果的单价: 有 4 种水果, 苹果(apples)、梨(pears)、桔子(oranges)和葡萄(grapes), 单价分别是 3.00 元/公斤, 2.50 元/公斤, 4.10 元/公斤和 10.20 元/公斤。在屏幕上显示以下菜单(编号和选项), 用户可以连续查询水果的单价, 当查询次数超过 5 次时, 自动退出查询; 不到 5 次时, 用户可以选择退出。当用户输入编号 1~4, 显示相应水果的单价(保留一位小数); 输入 0, 退出查询; 输入其他编号, 显示价格为 0。试编写相应程序。

```

[1] apples
[2] pears
[3] oranges
[4] grapes
[0] Exit

```

解答:

```

#include <stdio.h>

int main (void)
{
    int choice, i;
    double price;

    printf ("[1] apple\n");
    printf ("[2] pear\n");
    printf ("[3] orange\n");
    printf ("[4] grape\n");
    printf ("[0] exit\n");
    for (i = 1; i <= 5; i++){
        scanf ("%d", &choice);
        if (choice == 0)

```

```

        break;
    else{
        switch (choice){
            case 1: price = 3.00; break;
            case 2: price = 2.50; break;
            case 3: price = 4.10; break;
            case 4: price = 10.20; break;
            default: price = 0; break;
        }

        printf ("price = %0.2f\n", price);
    }
}

return 0;
}

```

3-9 请读者重新编写例 3-4 的程序，要求使用嵌套的 if-else 语句，并上机运行。

解答：

```

#include<stdio.h>
int main(void)
{
    double x, y;

    scanf("%lf", &x);
    if (x > 15){
        y = 2.5 * x - 10.5;
    }
    else{
        if (x < 0)
            y = 0;
        else
            y = 4 * x/3;
    }
    printf ("f(%0.2f)=%0.2f\n", x, y);

    return 0;
}

```

3-10 在例 3-12 中，改写 if 语句前， $y = x + 1$; 和 $y = x + 2$; 两条语句的执行条件是什么？改写后呢？

解答：

改写前： $y = x + 1$ 的执行条件是 $x < 1$; $y = x + 2$ 的执行条件是 $1 \leq x < 2$ 。

改写后： $y = x + 1$ 的执行条件是 $x < 1$; $y = x + 2$ 的执行条件是 $2 \leq x$ 。

3.3.2 习题参考答案

一. 选择题

1	2	3	4	5	6	7	8
C	B	A	D	D	C	A	B

二. 填空题

1	输入 32, 输出 32 输入 58, 输出 585858 x % 2 == 0	2	(x > 10 && x < 100) x < 0
3	first == 1 first = 0; printf("%d", x); max = a; max = c; b > c max = c;	4	<u>onetwo</u>

三. 程序设计题

1. 比较大小：输入 3 个整数，按从小到大的顺序输出。试编写相应程序。

解答：

```
#include <stdio.h>
int main (void)
{
    int a, b, c, t;

    scanf ("%d%d %d ", &a, &b, &c);
    if (a > b){
        t = a; a = b; b = t;
    }
    if (a > c){
        t = a; a = c; c = t;
    }
    if (b > c){
        t = b; b = c; c = t;
    }
    printf ("%d->%d->%d\n", a, b, c);

    return 0;
}
```

2. 高速公路超速处罚：按照规定，在高速公路上行使的机动车，超出本车道限速的 10% 则处 200 元罚款；若超出 50%，就要吊销驾驶证。请编写程序根据车速和限速自动判别对该机动车的处理。

解答：

```
#include <stdio.h>
```

```

int main (void)
{
    double limit, rate, speed;

    scanf ("%lf%lf", &speed, &limit);
    rate = 100 * (speed - limit) / limit;
    if (rate < 10){
        printf("OK\n");
    }
    else if (rate < 50){
        printf("Exceed %.0f%%. Ticket 200\n", rate);
    }
    else{
        printf("Exceed %.0f%%. License Revoked\n", rate);
    }

    return 0;
}

```

3. 出租车计价：某城市普通出租车收费标准如下：起步里程为 3 公里，起步费 10 元；超起步里程后 10 公里内，每公里 2 元；超过 10 公里以上的部分加收 50%的回空补贴费，即每公里 3 元；营运过程中，因路阻及乘客要求临时停车的，按每 5 分钟 2 元计收(不足 5 分钟则不收费)。运价计费尾数四舍五入，保留到元。编写程序，输入行驶里程（公里）与等待时间（分钟），计算并输出乘客应支付的车费（元）。

解答：

```

#include <stdio.h>
int main (void)
{
    int time;
    double distance, fee;

    scanf ("%lf%d", &distance, &time);
    if (distance <= 3){
        fee = 10;
    }
    else if (distance <= 10){
        fee = 10 + (distance - 3) * 2;
    }
    else{
        fee = 10 + 7 * 2 + (distance - 10) * 2 * 1.5;
    }
    fee = fee + time / 5 * 2;
    printf ("%f\n", fee);
}

```

```

    return 0;
}

```

4. 统计学生成绩：输入一个正整数 n ，再输入 n 个学生的成绩，统计五分制成绩的分布。百分制成绩到五分制成绩的转换规则：大于或等于 90 分为 A，小于 90 分且大于或等于 80 分为 B，小于 80 分且大于或等于 70 为 C，小于 70 分且大于或等于 60 为 D，小于 60 分为 E。试编写相应程序。

解答：

```

#include <stdio.h>
int main (void)
{
    int ca, cb, cc, cd, ce, grade, i, n;

    scanf("%d", &n);
    ca = cb = cc = cd = ce = 0;
    for (i = 0; i < n; i++) {
        scanf("%d", &grade);
        switch(grade / 10) {
            case 10: case 9: ca++; break;
            case 8: cb++; break;
            case 7: cc++; break;
            case 6: cd++; break;
            default: ce++; break;
        }
    }
    printf ("%d %d %d %d %d\n", ca, cb, cc, cd, ce);

    return 0;
}

```

5. 三角形判断：输入平面上任意三个点的坐标(x1,y1)、(x2,y2)、(x3,y3)，检验它们能否构成三角形。如果这 3 个点能构成一个三角形，输出周长和面积（保留 2 位小数）；否则，输出 “Impossible”。试编写相应程序。

解答：

```

#include <stdio.h>
#include <math.h>
int main (void)
{
    double x1, y1, x2, y2, x3, y3;
    double d12, d13, d23;
    double area, s;

    scanf ("%lf%lf", &x1, &y1);
    scanf ("%lf%lf", &x2, &y2);

```

```

scanf ("%lf%lf", &x3, &y3);
d12 = sqrt ((x1 - x2) * (x1 - x2) + (y1 - y2) * (y1 - y2));
d13 = sqrt ((x1 - x3) * (x1 - x3) + (y1 - y3) * (y1 - y3));
d23 = sqrt ((x2 - x3) * (x2 - x3) + (y2 - y3) * (y2 - y3));
if ((d12 + d13 > d23) && (d13 + d23 > d12) && (d12 + d23 > d13)){
    s=(d12 + d13 + d23) / 2;
    area = sqrt (s * (s - d12) * (s - d13) * (s - d23));
    printf ("L = %.2lf, A = %.2lf\n", s * 2.0, area);
}
else
    printf ("Impossible\n");

return 0;
}

```

3.4 实验指导教材参考答案

一、调试示例

统计 MOOC 证书：学生修读程序设计 MOOC，60 分及以上获得合格证书，85 分及以上获得优秀证书，不到 60 分则没有证书。输入一个非负整数 n，再输入 n 个学生的 MOOC 成绩，统计优秀、合格证书的数量，以及没有获得证书的数量。

解答：

```

#include <stdio.h>
int main(void)
{
    int cnt_a, cnt_f, cnt_p, i, n, score;

    printf("Enter n(n>0): ");
    scanf("%d", &n);
    cnt_a = cnt_p = cnt_f = 0;
    for(i = 1; i <= n; i++){
        scanf ("%d", &score);
        if(score >= 80){
            cnt_a++;
        }else if(score >= 60){
            cnt_p++;
        }else{
            cnt_f++;
        }
    }
    printf("%d %d %d\n", cnt_a, cnt_p, cnt_f);

    return 0;
}

```


}

二、基础编程题

(1) 计算符号函数的值：输入一个整数 x ，计算并输出符号函数 $\text{sign}(x)$ 的值。试编写相应程序

$$y = \text{sign}(x) = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases}$$

解答：参见练习 3-2。

(2) 比较大小：输入 3 个整数 n_1 、 n_2 和 n_3 ，将这 3 个数按从小到大的顺序输出。试编写相应程序。

解答：参见习题程序设计第 1 题。

(3) 统计英文字母、空格或换行、数字字符：输入一个正整数 n ，再输入 n 个字符，统计其中英文字母、空格或回车、数字字符和其他字符的个数。试编写相应程序。

解答：参见练习 3-4。

(4) 查询水果价格：有苹果 (apple)、梨 (pear)、橘子 (orange) 和葡萄 (grape) 4 种水果，单价分别是 3.00 元/千克，2.50 元/千克，4.10 元/千克和 10.20 元/千克。在屏幕上显示以下菜单 (编号和选项)，用户可以连续查询水果的单价，当查询次数超过 5 次时，自动退出查询；不到 5 次时，用户可以选择退出。用户输入编号 `choice`，输入 1 ~ 4，显示相应水果的单价 (保留 1 位小数)；输入 0，退出查询；输入 0 ~ 4 之外的其他编号，显示价格为 0。试编写相应程序。

解答：参见练习 3-8。

(5) 计算个人所得税：假设个人所得税为：税率 * (工资 - 1600)。请编写程序计算应缴的所得税，其中税率定义为：

- 当工资不超过 1600 时，税率为 0；
- 当工资在区间 (1600, 2500] 时，税率为 5%；
- 当工资在区间 (2500, 3500] 时，税率为 10%；
- 当工资在区间 (3500, 4500] 时，税率为 15%；
- 当工资超过 4500 时，税率为 20%。

解答：

```
#include <stdio.h>

int main(void)
{
    double salary, rate;

    scanf("%lf", &salary);
    if (salary > 4500.0){
        rate = 0.20;
```

```

    }
    else if (salary > 3500.0){
        rate = 0.15;
    }
    else if (salary > 2500.0){
        rate = 0.10;
    }
    else if (salary > 1600.0){
        rate = 0.05;
    }
    else{
        rate = - 0.0;
    }
    printf("%.2f\n", (salary - 1600) * rate);

    return 0;
}

```

(6) 统计学生成绩：输入一个正整数 n ，再输入 n 个学生的百分制成绩，统计各等级成绩的个数。成绩等级分为五级，分别为 A (90 ~ 100)、B (80 ~ 89)、C (70 ~ 79)、D (60 ~ 69) 和 E (0 ~ 59)。试编写相应程序。

解答：参见习题程序设计第 4 题。

三、改错题

输出三角形面积和周长：输入三角形的 3 条边 a 、 b 、 c ，如果能构成一个三角形，输出面积 $area$ 和周长 $perimeter$ （保留 2 位小数）；否则，输出 “These sides do not correspond to a valid triangle”。（源程序 error03_2.cpp）

在一个三角形中，任意两边之和大于第 3 边。三角形面积计算公式：

$$area = \sqrt{s(s-a)(s-b)(s-c)}$$

其中， $s = (a+b+c)/2$

解答：

```

#include <stdio.h>
#include <math.h>
int main (void)
{
    double a, b, c;
    double area, perimeter, s;

    printf("Enter 3 sides of the triangle: ");
    scanf ("%lf%lf%lf", &a, &b, &c);
    if ((a + b > c) && (b + c > a) && (a + c > b)){
        s = (a + b + c) * 1.0 / 2;
        area = sqrt(s * (s-a) * (s-b) * (s-c));
    }
}

```

```

        perimeter = a + b + c;
        printf ("area = %.2f; perimeter = %.2f\n",area, perimeter);
    }
    else {
        printf ("These sides do not correspond to a valid triangle\n");
    }

    return 0;
}

```

四、拓展编程题

(1) 三天打鱼两天晒网：中国有句俗语叫“三天打鱼两天晒网”。假设某人从某天起，开始“三天打鱼两天晒网”，问这个人在以后的第 N 天中是“打鱼”还是“晒网”？试编写相应程序。

解答：

```

#include <stdio.h>
int main(void)
{
    int m, n;

    scanf ("%d", &n);
    m = n % 5;
    if ((m >= 1) && (m < 4)){
        printf ("Fishing in day %d\n", n);
    }
    else{
        printf ("Drying in day %d\n", n);
    }

    return 0;
}

```

(2) 计算油费：假设现在 90 号汽油 6.95 元/升、93 号汽油 7.44 元/升、97 号汽油 7.93 元/升。为吸引顾客，某自动加油站推出了“自助服务”和“协助服务”两个服务等级，分别可得到 5%和 3%的折扣。请编写程序，输入顾客的加油量 a，汽油品种 b(90、93 或 97)和服务类型 c (m 为自助服务，e 为协助服务)，计算并输出应付款（保留小数点后 2 位）。

解答：

```

#include <stdio.h>
int main (void)
{
    int a, b;
    char c ;
    double price, money;

    scanf ("%d%d", &a, &b);

```

```

getchar ();
c = getchar ();
if (b == 90){
    price = 6.95;
}
else if (b == 93){
    price = 7.44;
}
else
    price = 7.93;
if(c == 'm')
    money = a * price * 0.95;
else
    money = a * price * 0.97;
printf (".2f\n", money);

return 0;
}

```

(3) 求一元二次方程的根：输入参数 a、b、c，求一元二次方程 $ax^2 + bx + c = 0$ 的根。

- ①如果方程有两个不相等的实数根，则每行输出一个根，先大后小；
- ②如果方程有两个不相等复数根，则每行按照格式“实部+虚部 i”输出一个根，先输出虚部为正的，后输出虚部为负的；
- ③如果方程只有一个根，则直接输出此根；
- ④如果系数都为 0，则输出"Zero Equation"；
- ⑤如果 a 和 b 为 0，c 不为 0，则输出"Not An Equation"。

解答：

```

#include <stdio.h>
#include <math.h>
#define eps 0.0000000001
int main (void)
{
    double a, b, c, d;

    scanf ("%lf%lf%lf", &a, &b, &c);
    d = b * b - 4 * a * c;
    if (fabs (a) < eps){
        if (fabs (b) < eps){
            if (fabs (c) < eps){
                printf ("Zero Equation\n");
            }
            else{
                printf ("Not An Equation\n");
            }
        }
    }
}

```

```

    }
    else{
        printf ("%0.2f\n", -c / b);
    }
}
else{
    if (fabs (d) < eps){
        printf ("%0.2f\n", -b / (a+a) );
    }
    else if (d > 0){
        printf ("%0.2f\n", (-b + sqrt (d)) / (2 * a));
        printf ("%0.2f\n", (-b - sqrt (d)) / (2 * a));
    }
    else if (fabs (b) < eps){
        printf ("0.00+%0.2fi\n", sqrt (-d) / (2 * a));
        printf ("0.00-%0.2fi\n", sqrt (-d) / (2 * a));
    }
    else{
        printf ("%0.2f+%0.2fi\n", -b / (2 * a), sqrt (-d) / (2 * a));
        printf ("%0.2f-%0.2fi\n", -b / (2 * a), sqrt (-d) / (2 * a));
    }
}

return 0;
}

```