

C Programming Language

- SWITCH
- Ternary operator

Dr. Mostafa A. Elhosseini

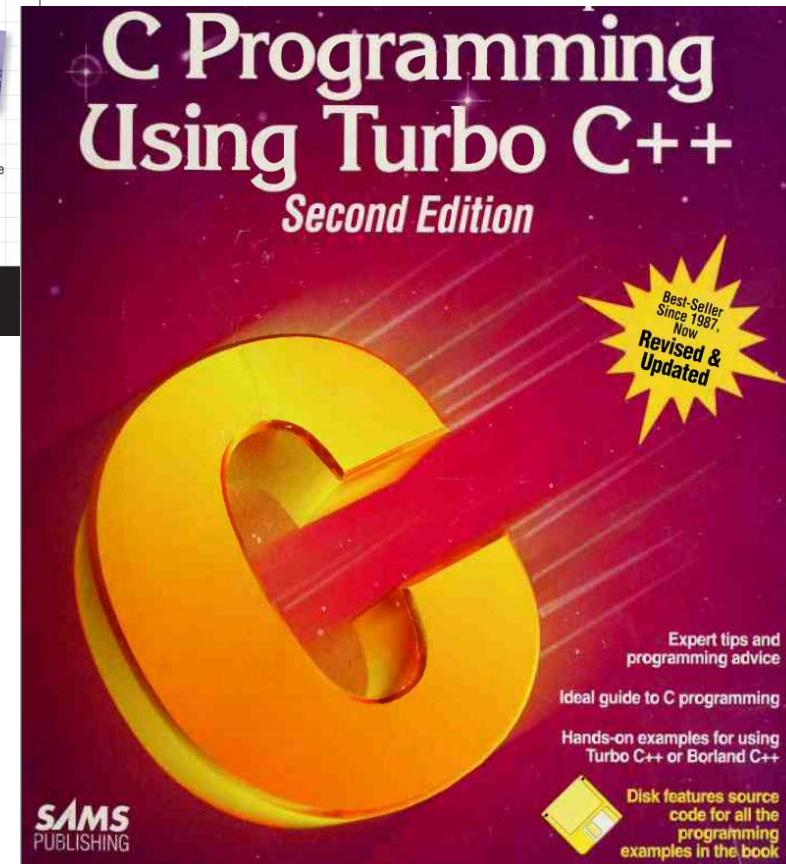
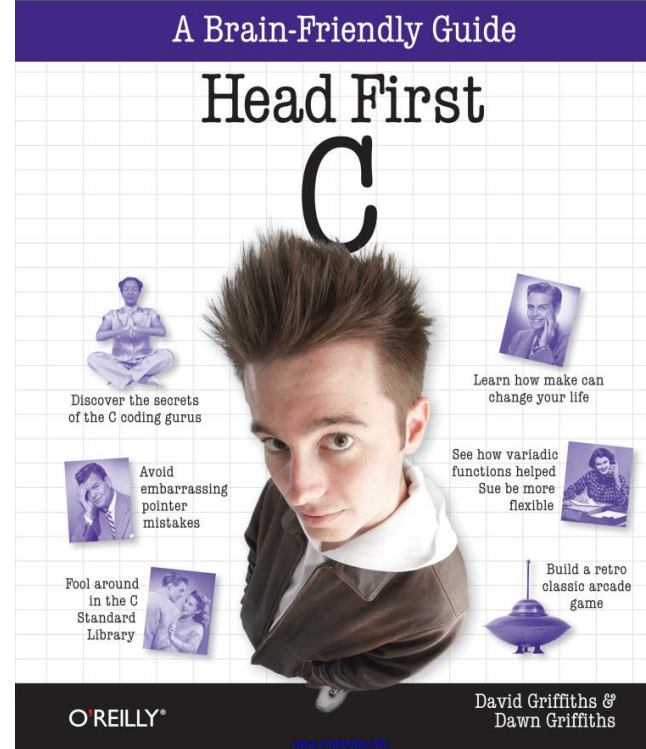
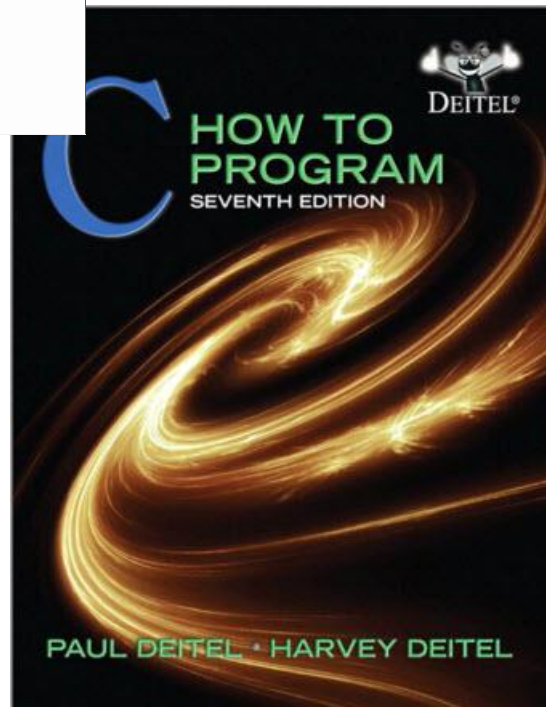
SECOND EDITION

THE



PROGRAMMING LANGUAGE

BRIAN W. KERNIGHAN
DENNIS M. RITCHIE



Revise

✂ Conditional statements

- if
- if – else
- else if

Agenda

✂ switch statement

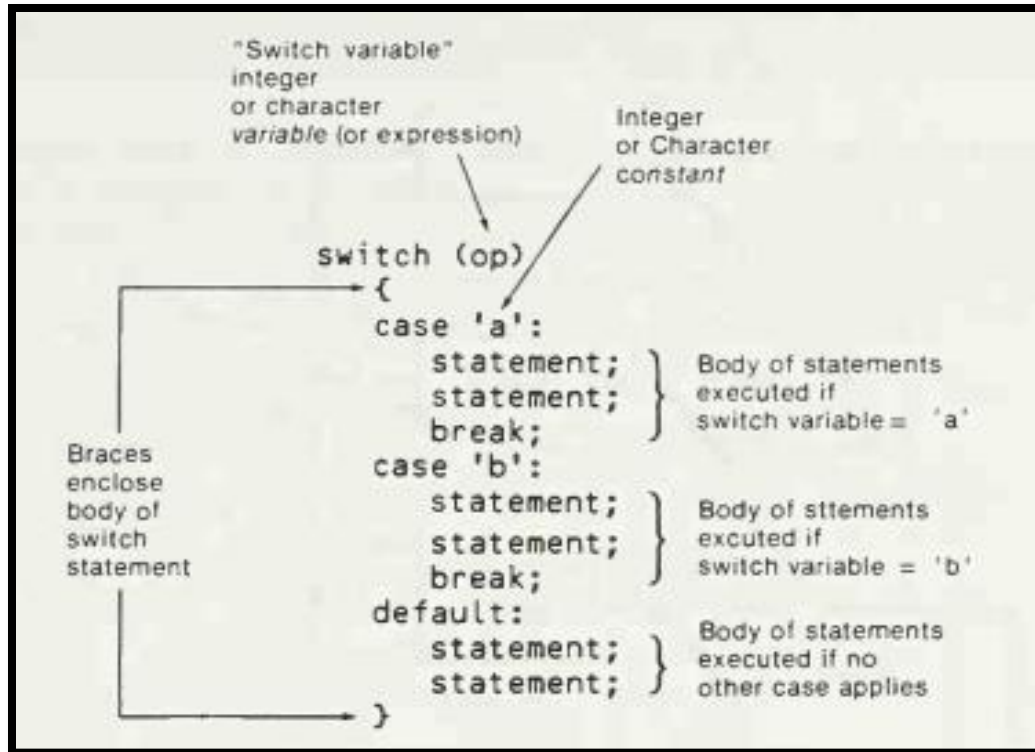
- How it works
- case
- default
- break

✂ Ternary operator

Switch

- ✧ Switch statement is **multiway branch** based on the value of a control expression
 - To facilitate **selecting** between **multiple alternatives**
 - Test whether an expression matches one of a number of constant integer values (int – char)
- ✧ The **switch** statement is similar to the **else -if** construct but has more flexibility and a clearer format.
 - The **switch** statement can not handle ranges
- ✧ It is analogous to the case statement in **Pascal** and the **select case** statement in the **newer versions of BASIC**

Switch



How it works

- ✧ The **control expression is evaluated**
- ✧ If the **value of the control** expression is equal to that of the **constant expression** in some case label belonging to the switch statement
 - Program control is transferred to the point indicated by the case label
- ✧ If the value of the control expression is not equal to any case label, and there is no default label
 - Program control is transferred to whatever follows the switch statement

```
1  #include <stdio.h>
2  int main()
3  {
4      char a;
5      puts("Your choice (1,2,3): ");
6      scanf("%c",&a);
7      switch(a)
8      {
9          case '1':
10             puts("Excellent choice");
11             break;
12          case '2':
13             puts("this is the most common choice");
14             break;
15          case '3':
16             puts("I questions your judgment");
17             break;
18          default:
19             puts("That is not a valid choice");
20
21     }
22     return 0;
23 }
```

"E:\Courses\Taibah\Programming essential using C\Code\test\bin\De

Your choice (1,2,3):

2

this is the most common choice

Process returned 0 (0x0) exec

Press any key to continue.

break

- ✧ The break statement **causes an immediate exit from the switch**
 - Execution falls through to the next unless you take explicit action to escape
 - break
 - return
- ✧ If no break statement is used following a case, control will fall through to the next case.
 - Falling through cases is a mixed blessing.
 - On the positive side, it allows several cases to be attached to a single action
- ✧ As a **matter of good form, put a break after the last case**
 - Even though it is **logically unnecessary**
 - Some day **when another case gets added at the end**, this bit of defensive programming will save you

```
1  #include <stdio.h>
2  int main()
3  {
4      int x;
5      puts("Your choice (0,1,2,3): ");
6      scanf("%d", &x);
7      switch(x)
8      {
9          case 0:
10             str = "none";
11             break;
12          case 1:
13             str = "single";
14             break;
15          case 2:
16             str = "pair";
17             break;
18          default:
19             str = "many";
20
21     }
22     return 0;
23 }
```

case with break

```
1  #include <stdio.h>
2  int main()
3  {
4      int x;
5      puts("Your choice (0,1): ");
6      scanf("%d",&x);
7      switch(x)
8      {
9          case 0:
10             printf("A,");
11          case 1:
12             printf("B, ");
13             break;
14          default:
15             printf("C");
16
17      }
18      return 0;
19  }
20
```

"E:\Courses\Taibah\Programming essential using C\Code\te

Your choice (0,1):
0
A,B,
Process returned 0 (0x0)
Press any key to continue.

case with break

```
1  #include <stdio.h>
2  int main()
3  {
4      int x;
5      puts("Enter a number : (1,2,3,4) ");
6      scanf("%d",&x);
7      switch(x)
8      {
9          case 1: printf("*");
10         case 2: printf("**");
11         case 3: printf("***");
12         case 4: printf("****");
13     }
14     return 0;
15 }
16
```

default

- ✧ The case labeled **default** is executed if none of the other cases are satisfied
- ✧ A default is optional
 - If **it is not there** and if **none of the cases match**, no **action** at all takes place
 - **Cases** and the **default** clauses can occur in any order

Examples

- Calculator example

```
1  #include <stdio.h>
2  int main()
3  {
4      float x,y;
5      char op;
6      puts("Enter two numbers and operators: (x op y) ");
7      scanf("%f %c %f",&x,&op,&y);
8      switch(op)
9      {
10     case '+':
11         printf("sum = %f",x+y);
12         break;
13     case '-':
14         printf("sub = %f",x-y);
15         break;
16     case '*':
17         printf("Mul = %f",x*y);
18         break;
19     case '/':
20         printf("sub = %f",x/y);
21         break;
22     default:
23         printf("Illegal operator");
24     }
25     return 0;
```

"E:\Courses\Taibah\Programming essential using C\Code\test\bin\Debug\test.exe"

```
Enter two numbers and operators: (x op y)
7 / 3
sub = 2.333333
Process returned 0 (0x0)   execution time :
Press any key to continue.
```

Many repeated cases

```
9 {
10 case '+':
11     printf("sum = %f", x+y);
12     break;
13 case '-':
14     printf("sub = %f", x-y);
15     break;
16 case '*':
17 case 'x':
18     printf("Mul = %f", x*y);
19     break;
20 case '/':
21     printf("sub = %f", x/y);
22     break;
23 default:
24     printf("Illegal operator");
25 }
26 return 0;
27 }
28
```

"E:\Courses\Taibah\Programming essential using C\Code\test\bin\Debug\test.exe"

```
Enter two numbers and operators: (x op y)
4 x 8
Mul = 32.000000
Process returned 0 (0x0)   execution time :
Press any key to continue.
```

Example

```
int c, i, nwhite, nother, ndigit[10];

nwhite = nother = 0;
for (i = 0; i < 10; i++)
    ndigit[i] = 0;
while ((c = getchar()) != EOF) {
    switch (c) {
        case '0': case '1': case '2': case '3': case '4':
        case '5': case '6': case '7': case '8': case '9':
            ndigit[c-'0']++;
            break;
        case ' ':
        case '\n':
        case '\t':
            nwhite++;
            break;
        default:
            nother++;
            break;
    }
}
```

Examples

- Circle Area
- Circle circumference

```
4  #include <math.h>
5  #define PI 3.14159
6  int main()
7  {
8      float r;
9      char op;
10     puts("for area use \' A\");
11     puts("for Circumference use \' C\");
12     scanf("%c",&op);
13     puts("Circle Radius: ");
14     scanf("%f",&r);
15     switch(op)
16     {
17     case 'A':
18     case 'a':
19         printf("area = %f",PI*pow(r,2));
20         break;
21     case 'C':
22     case 'c':
23         printf("circum = %f",2*PI*r);
24         break;
25     default:
26         printf("unknown option chosen");
27     }
```

Conditional operator

- Ternary operator
- **expr1 ? Expr2 : expr3**
- The expression expr1 is evaluated first
 - If it is non-zero (true), then the **expression expr2 is evaluated**, and that is the **value of the conditional expression**
 - **Otherwise** expr3 is evaluated, and that is the value
- **z = (a > b) ? a: b;**
- **Conditional expression is indeed an expression** , and it can be used wherever any other expression can be

```
1  #include <stdio.h>
2  #include <math.h>
3  int main()
4  {
5      float x,y,max;
6      puts("Enter two numbers to get the maximum: (x y) ");
7      scanf("%f %f",&x,&y);
8      max = x>y? x: y;
9      printf("Maximum number: %f",max);
10     return 0;
11 }
12
```

"E:\Courses\Taibah\Programming essential using C\Code\test\bin\Debug\test.exe"

```
Enter two numbers to get the maximum: (x y)
4 15
Maximum number: 15.000000
Process returned 0 (0x0)   execution time :
Press any key to continue.
```

```
1  #include <stdio.h>
2  #include <math.h>
3  int main()
4  {
5      float x,y;
6      puts("Enter two numbers to get the maximum: (x y) ");
7      scanf("%f %f",&x,&y);
8      printf("%f %c %f",x, (x>y? '>': '<'), y);
9      return 0;
10 }
11
```

"E:\Courses\Taibah\Programming essential using C\Code\test\bin\Debug\test.exe"

Enter two numbers to get the maximum: (x y)

10 -7

10.000000 > -7.000000

Process returned 0 (0x0) execution time : 4.563 s

Press any key to continue.