

Chapter 3

Flow of Control

SPRING 2003

COP 3223

1

Control Structures

All programs written in terms of 3 control structures:

- Sequence structures: Built into C. Programs executed sequentially by default.
- Selection structures: C has three types: **if**, **if/else**, and **switch**
- Repetition structures: C has three types: **while**, **do/while** and **for**

SPRING 2003

COP 3223

2

Relational Operators

- The relational operators are **<**, **>**, **<=**, and **>=**.
- They take **2** expressions as operands and yield either the `int` value **0** (false) or the `int` value **1** (true).
- Valid

a < 3
a > b

Invalid

a = < b
a < = b

- Examples: Assume a = 1, b=2.

<u>Expression</u>	<u>Value</u>
a <= b	1
a < b-5	0
a + 10 / b <= -3 + 8	0

SPRING 2003

COP 3223

3

Equality Operators

- The equality operators are **==** and **!=**.
- Yield either the `int` value **0** or the `int` value **1**.
- Valid

x != -2.77
x + 2.0 != 3.3/z
ch == '*'

Invalid

x == y-1
x != 44
ch = '*'

- Examples: Assume a=1, b=2, ch = 'A'

<u>Expression</u>	<u>Value</u>
a == b	0
a != b	1
ch < 'B'	1
a+b == -2 * 3	0

SPRING 2003

COP 3223

4

Equality Operators

- Note carefully that the two expressions
`a == b` and `a = b`
are visually similar.
- The expression `a == b` is a test for equality
and `a = b` is an assignment expression.

SPRING 2003

COP 3223

5

Logical Operators

- The logical operators are `&&`, `||`, and `!`.
- Expressions connected by `&&` or `||` are evaluated left to right.
- Logical negation: `!`

<u>Value of expression</u>	<u>!expression</u>
zero	1
nonzero	0

- Examples

<u>Expression</u>	<u>Value</u>
<code>!5</code>	0
<code>!!5</code>	1
<code>!(6 < 7)</code>	0
<code>!6 < 7</code>	1
<code>!(3-4)</code>	0

SPRING 2003

COP 3223

6

Logical Operators

<u>a</u>	<u>b</u>	<u>a && b</u>	<u>a b</u>
zero	zero	0	0
zero	nonzero	0	1
nonzero	zero	0	1
nonzero	nonzero	1	1

SPRING 2003

COP 3223

7

Examples

- Given declarations:

```
int a = 3, b = 3, c = 3;
double x = 0.0, y = 2.5;
char ch = 'g'
```

<u>Expression</u>	<u>Value</u>
<code>!(a < b) && c</code>	1
<code>ch >= 'a' && ch <= 'z'</code>	1
<code>x a && b - 3</code>	0
<code>a < b && x < y</code>	0
<code>a < b x < y</code>	1

- The precedence of `&&` is higher than `||`, but both operators are of lower precedence than all unary, arithmetic and relational operators. Their associativity is left to right.

SPRING 2003

COP 3223

8

Short-Circuit Evaluation

- For the expressions that contain the operands of `&&` and `||`, the expression process stops as soon as the outcome true or false is known.
- Suppose `expr1` is 0.
`expr1 && expr2 = 0` (expr2 will not be evaluated.)
- Suppose `expr1` is nonzero.
`expr1 || expr2 = 1` (expr2 will not be evaluated.)

SPRING 2003

COP 3223

9

Compound and Empty Statements

- ```

{
 a = 1;
 {
 b = 2;
 c = 3;
 }
}

```
- ```

a = b;
a + b + c;
;
printf("Hello");

```

SPRING 2003

COP 3223

10

The `if` Selection Structure

- Selection structure:
 - Used to choose among alternative courses of action
 - e.g:

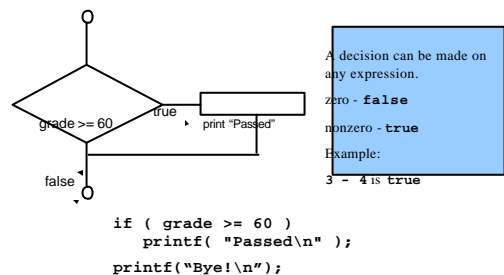
*If student's grade is greater than or equal to 60
Print "Passed"*
- If condition **true**
 - Print statement executed and program goes on to next statement
 - If **false**, print statement is ignored and the program goes onto the next statement

SPRING 2003

COP 3223

11

if Statement



SPRING 2003

COP 3223

12

```

/* Determines if a number is even */
#include <stdio.h>

int main(void)
{
    int value;

    printf("Enter a number.\n");
    scanf("%d",&value);
    if (value % 2 == 0)
        printf("\n%d is an even number.\n");
    return 0;
}

```

```

Enter a number.
10
10 is an even number.
Enter a number.
15

```

SPRING 2003

COP 3223

13

```

/* An example program using a compound statement in an if
statement */
#include <stdio.h>

int main ( )
{
    int value1, value2, max=0;

    printf("Enter two values:\n");
    scanf("%d%d", &value1, &value2);
    if (value1 > value2) {
        max = value1;
        printf("Value1 is greater than value2. \n");
    }
    printf("%d\n", max);
    return 0;
}

```

```

Enter two values:
10 5
Value1 is greater than value2.
10
Enter two values:
10 20
0

```

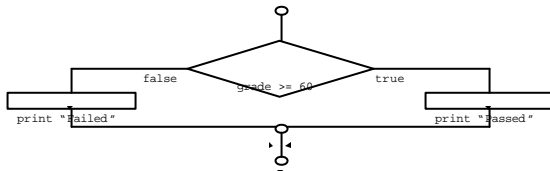
SPRING 2003

COP 3223

14

if-else statement

- Flow chart of the **if/else** selection structure



```

if ( grade >= 60 )
    printf( "Passed\n");
else
    printf( "Failed\n");

```

SPRING 2003

COP 3223

15

```

/* Determines the larger of two numbers */
#include <stdio.h>

int main (void)
{
    int v1, v2, larger;

    printf("Enter two numbers.\n");
    scanf("%d%d", &v1, &v2);

    if (v1 > v2)
        larger = v1;
    else
        larger = v2;

    printf("%d is the larger number.\n", larger);
    return 0;
}

```

```

Enter two numbers.
10 43
43 is the larger number.
Enter two numbers.
56 11
56 is the larger number.

```

SPRING 2003

COP 3223

16

Examples

- Compound statement:

```
if ( grade >= 60 )
    printf( "Passed.\n" );
else {
    printf( "Failed.\n" );
    printf( "You must take this course again.\n" );
}
```

- Dangling else: an else attaches to the nearest if .

```
if (a == 10)
    if (b == 20)
        printf( "****\n" );
else printf( "###\n" );
```

SPRING 2003

COP 3223

17

Problem Solving

```
// Find the minimum of three values.
#include <stdio.h>

int main()
{
    int a, b, c, min;

    printf("Enter three numbers:");
    scanf("%d%d%d", &a,&b,&c);

    if (a < b)
        min = a;
    else min = b;
    if (c < min)
        min = c;

    printf("The minimum value is %d\n", min);
    return 0;
}
```

SPRING 2003

COP 3223

18

Nested if/else structures

- Test for multiple cases by placing **if/else** selection structures inside **if/else** selection structures.
- Once condition is met, the rest of statements skipped.
- Its general form is:

```
if (expr1)
    statement1
else if (expr2)
    statement2
else if (expr3)
    statement3
.....
else if (exprN)
    statementN
else
    default statement
next statement
```

SPRING 2003

COP 3223

19

Nested if's

```
if (grade >= 90)
    printf("A");
else if (grade >= 80)
    printf("B");
else if (grade >= 70)
    printf("C");
else if (grade >= 60)
    printf("D");
else
    printf("F");
```

SPRING 2003

COP 3223

20

The Ternary Conditional Operator (?:)

cond ? expr1 : expr2

- Takes three arguments (condition, value if **true**, value if **false**)
- e.g:

```
printf("%s\n", grade >= 60 ? "Passed" : "Failed" );
```
- Or it could have been written:

```
grade >= 60 ? printf( "Passed\n" ) :  
printf( "Failed\n" );
```
- Or it could be used in an assignment statement:

```
letter = (grade >= 60) ? 'S' : 'U';
```

SPRING 2003

COP 3223

21

The switch Multiple-Selection Structure

• switch

- Useful when a variable or expression is tested for all the values it can assume and different actions are taken

• Format

- Series of **case** labels and an optional **default** case

```
switch ( a_variable ){  
    case value1 :  
        actions  
    case value2 :  
        actions  
    ...  
    default :  
        actions  
}
```

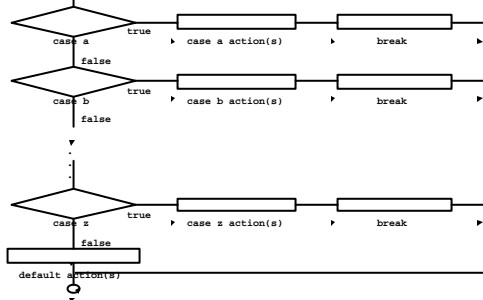
- **break**; exits from structure

SPRING 2003

COP 3223

22

The switch Multiple-Selection Structure



SPRING 2003

COP 3223

23

```
/*Counting letter grades */  
  
int grade;  
int aCount = 0, bCount = 0, cCount = 0,  
    dCount = 0, fCount = 0;  
  
printf("Enter the letter grade.\n" );  
scanf("%c",&grade);  
  
switch ( grade ) {  
    case 'A':  
        ++aCount;  
        break;  
    case 'B':  
        ++bCount;  
        break;  
    case 'C':  
        ++cCount;  
        break;  
    case 'D':  
        ++dCount;  
        break;  
    case 'F':  
        ++fCount;  
        break;  
    default:  
        /* catch all other characters */  
        printf( "Incorrect letter grade entered." );  
        printf( " Enter a new grade.\n" );  
        break;  
}
```

SPRING 2003

COP 3223

24

```

int main()
{
    int month, year, days, leapyear;

    printf("Enter a month and a year:");
    scanf("%d%d", &month, &year);
    if (((year % 4 == 0) && (year % 100 != 0)) || (year % 400 == 0) )
        leapyear = 1;
    else
        leapyear = 0;
    switch (month){
        case 9 :
        case 4 :
        case 6 :
        case 11: days=30;
                break;
        case 2 : days = (leapyear == 1)? 29: 28;
                break;
        default :
                days = 31;
    }
    printf("There are %d days in that month in that year.\n", days);
}

```

SPRING 2003

COP 3223

25