Relational Operators > < >= <=

- Evaluates to 1 for True and 0 for False
- $2 > 3 \Rightarrow 0$
- $3.1415 > 3 \Rightarrow 3.1415 > 3.0 \Rightarrow 1$
- $30 \leq 30 \Rightarrow 1$
- $10 < 9 \Rightarrow 0$
- $10 \geq 9 \Rightarrow 1$
- $-2 < -1 < 0 \Rightarrow 0$ (False??)
- $-2 < -1 < 0 \Rightarrow (-2 < -1) < 0 \Rightarrow 1 < 0 \Rightarrow 0$
Relational Operators == and !=

- 3 == 3 ⇒ 1
- 3 != 3 ⇒ 0
- 3.0 == 3 ⇒ 3.0 == 3.0 ⇒ 1 (be careful!)
- 1 < 2 == 2 < 3 ⇒ (1 < 2) == (2 < 3) ⇒ 1
- < > <= >= ⇒ 6th priority, left to right
- == != ⇒ 7th priority, left to right
- 2 + 3 < 2 * 3 ⇒ (2 + 3) < (2 * 3) ⇒ 1

Example: Conversion Table F2C

```cpp
void main () {
    float fahr, celsius;
    int lower, upper, step;

    lower = 0; /* lower limit of temperature table */
    upper = 300; /* upper limit */
    step = 20; /* step size */

    fahr = lower;
    while (fahr <= upper) {
        celsius = (5.0 / 9.0) * (fahr - 32.0);
        printf("%3.0f %6.1f\n", fahr, celsius);
        fahr = fahr + step;
    }
}
```
Example: A Nicer Table F2C

```c
#define LOWER 0
#define UPPER 300
#define STEP 20
#define TABLETOP "+-------------+
void main () {
    float fahr, celsius;
    fahr = LOWER;
    printf("Fahrenheit to Celsius\nTemperature Conversion\n\n");
    printf(TABLETOP);
    while (fahr <= UPPER) {
        celsius = (5.0 / 9.0) * (fahr - 32.0);
        printf ("| %3.0f %6.1f |\n", fahr, celsius);
        fahr += STEP;
    }
    printf(TABLETOP);
}
```

Programming Patterns

- Patterns are formed by several lines of code that could be distributed in the program text
- A pattern represent some typical task
- Once understood, a pattern can be used over and over
- Skilled programmers routinely use many patterns
Pattern: Processing a Table

```c
#define LOWER 0
#define UPPER 300
#define STEP 20
#define TABLETOP "+-------------+

main () {
    float fahr, celsius;
    fahr = LOWER;
    printf("Fahrenheit to Celsius
Temperature Conversion

");
    printf(TABLETOP);
    while (fahr <= UPPER) {
        celsius = (5.0 / 9.0) * (fahr - 32.0);
        printf("| %3.0f %6.1f |
", fahr, celsius);
        fahr += STEP;
    }
    printf(TABLETOP);
}
```

Some Typical Loops

- Counter controlled loop
  ```c
  n = 20;
  while(n > 0) { ...; --n;}
  ```

- Threshold controlled loop
  ```c
  x = 0; step = 10;
  while(x < 100) { ...; x += step;}
  ```

- Sentinel controlled loop
Example: Sentinel Control

/* Adds numbers until 0 is entered. Prints the sum. */
#include <stdio.h>
void main () {
    int sum = 0, nextnumber;
    /* pre-reading first number */
    printf("Number: ");
    scanf("%d", &nextnumber); /* read first number */
    while (nextnumber != 0) {
        sum += nextnumber;
        printf("Number: ");
        scanf("%d", &nextnumber);
    }
    printf ("Sum = %d\n", sum);
}

Pattern: Sentinel Input Processing

/* Add numbers until 0 is entered. Prints the sum. */
#include <stdio.h>
main () {
    int sum = 0, nextnumber;
    /* pre-reading first number */
    printf("Number: ");
    scanf("%d", &nextnumber); /* read first number */
    while (nextnumber != 0) {
        sum += nextnumber;
        printf("Number: ");
        scanf("%d", &nextnumber);
    }
    printf ("Sum = %d\n", sum);
}
Pattern: Summing a Sequence

/* Add numbers until 0 is entered. Prints the sum. */
#include <stdio.h>
main () {
    int sum = 0, nextnumber;
    /* pre-reading first number */
    printf("Number: ");
    scanf("%d", &nextnumber); /* read first number */
    while (nextnumber != 0) {
        sum += nextnumber;
        printf("Number: ");
        scanf("%d", &nextnumber);
    }
    printf ("Sum = %d\n", sum);
}

Do-while Loop

do
    statement
while (expression)

- The condition is checked after the execution
- The loop will be executed at least once
Do-while Loop

do {
    Statement-1
    ...
    Statement-K
} while (expression);
nextstatement

– If expression is not 0 (true) - back to statement-1
– If expression is 0 (false) - move to nextstatement
– I.e, while expression is true, do the loop

Flowchart of the do-while Loop
Example: Kids and Apples

void main () {
    int kids, apples, rounds = 0;
    printf("Kids?: "); scanf("%d", &kids);
    printf("Apples?: "); scanf("%d", &apples);
    do {
        apples -= kids;
        rounds++;
        printf("%d apples left after round %d\n",
               apples, rounds);
    } while (apples >= kids) ;
    printf ("Each kid got %d apples. %d apples left.\n",
            rounds, apples);
}

Example: Interest Table

void main() {
    int year = 1, how_many_years; /* years the capital stays in bank */
    float interest_rate; /* interest rate in percents */
    float capital; /* capital in dollars */
    float annual_interest; /* annual interest in dollars */
    printf("Startup capital ($$$CC): "); scanf("%f", &capital);
    printf("Interest rate (%% xx.xx): "); scanf("%f", &interest_rate);
    printf("How many years: ");
    scanf("%d", &how_many_years);
    /* Printing interest table */
    do {
        annual_interest = capital * interest_rate / 100;
        capital = capital + annual_interest;
        printf("Year %2d, capital %.2f\n", year, capital);
        year++;
    } while (year <= how_many_years);
    printf("New capital %.2f\n", capital);
}
Pattern: Analog of iterate Loop

void main() {
    int year = 1, how_many_years; /* years the capital stays in bank */
    float interest_rate; /* interest rate in percents */
    float capital; /* capital in dollars */
    float annual_interest; /* annual interest in dollars */

    printf("Startup capital ($$$\text{cc}): "); scanf("%f",&capital);
    printf("Interest rate (%% xx.xx): "); scanf("%f",&interest_rate);
    printf("How many years: ");
    scanf("%d", &how_many_years);

    /* Printing interest table */
    do {
        annual_interest = capital * interest_rate / 100;
        capital = capital + annual_interest;
        printf("Year %2d, capital \%.2f
", year, capital);
        year++;
    } while (year <= how_many_years);
    printf("New capital \%9.2f
", capital);
}

Foolproof Input with do

void main() {
    int year = 1, how_many_years; /* years the capital stays in bank */
    float interest_rate; /* interest rate in percents */
    float capital; /* capital in dollars */
    float annual_interest; /* annual interest in dollars */

    printf("Startup capital ($$$\text{cc}): "); scanf("%f",&capital);
    printf("Interest rate (%% xx.xx): "); scanf("%f",&interest_rate);
    printf("How many years (positive integer): ");
    scanf("%d", &how_many_years);
    } while ( how_many_years <= 0);
    do {
        annual_interest = capital * interest_rate / 100;
        capital = capital + annual_interest;
        printf("Year %2d, capital \%.2f
", year, capital);
        year++;
    } while (year <= how_many_years);
    printf("New capital \%9.2f
", capital);
}
Pattern: Foolproof Input with \textit{do}

\begin{verbatim}
do {
  \texttt{<request input>}
} while ( \texttt{<input is incorrect> } );
\end{verbatim}

\begin{itemize}
  \item \textbf{Examples:}
  \begin{verbatim}
do {
    printf("Enter an even number: ");
    scanf("%d", \&number);
} while ( number \% 2 );
/* same as number \% 2 == 1 */
\end{verbatim}
\end{itemize}

Before Next Lecture:

\begin{itemize}
  \item Do reading assignment (quiz!)
    \begin{itemize}
      \item Perry: Ch. 11: Testing Data; Ch. 14
      \item Run Classroom Examples
    \end{itemize}
  \item Exercise: Good-looking table for kids/apple problem with input check
  \item Exercise: Good-looking table for Celsius to Fahrenheit conversion
  \item Homework 6 (due 10/26/2004) - Loops
  \item Check yourself by working with KnowledgeTree and WADEIn system
\end{itemize}