Variables

* Allows us to refer to values by name
* Must be declared before first use
* Declaration must provide type, name, initial value (optional)
Examples:

```c
int a; /* 2's complement integer */
unsigned int b; /* unsigned integer */
float volume, height; /* floating-point, 32 bits */
double pi, epsilon; /* floating-point, 64 bits */
char c = 'A'; /* ASCII representation of character */
```
* Variable scope defined by location of declaration:
  1) Local: if defined inside some function
  2) Global: defined outside of any function

Example:

```c
#include <stdio.h>

float pi = 3.1415; /* Global variable */

int main
{
    int i=1; /* Local variable */
    printf("pi=%f, i=%d", pi, i);
    return 0;
}
```
Operators

* Assignment (=): value on the right will be assigned to variable on the left. Data conversion will take place if necessary.

```c
int i;
float f;
i = 2;    /* i holds value of 2 */
f = i;    /* f holds value of 2 now */
```
* Bitwise operators:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOT</td>
<td>~</td>
</tr>
<tr>
<td>AND</td>
<td>&amp;</td>
</tr>
<tr>
<td>OR</td>
<td></td>
</tr>
<tr>
<td>XOR</td>
<td>^</td>
</tr>
<tr>
<td>LEFT-SHIFT</td>
<td>&lt;&lt;</td>
</tr>
<tr>
<td>RIGHT-SHIFT</td>
<td>&gt;&gt;</td>
</tr>
</tbody>
</table>

Examples:

```
// x and y have bits flipped
y = ~x;

// shift-left y two positions
z = y<<2;

// (y&z) | x
w = (y&z) | x;
```
* Arithmetic operators:

Addition: +               Division: /
Subtraction: −               Remainder: %
Multiplication: *

* Example:

z = (x+y)*(w%m);

f = x+y*z+w;  /* Be careful what you mean! */

g = (x+y)*(z+w);  /* Use parenthesis to make
clear evaluation order */
Control structures

* Sequential construct

Flow diagram

1. Statement 1
2. Statement 2
3. Statement 3

Example

```c
/* get equation coefficients */
printf("Enter a, b, and c: ");
scanf("%f %f %f", &a, &b, &c);
printf("Solving equation \(x^2+\) \(ax+c=0\)\n", a, b, c);

/* compute solution */
D = b * b - 4 * a * c;  /* compute determinant */
```
* Conditional construct

Flow diagram

Example

```c
if (D == 0)
    x1 = -b / (2 * a);
```
Flow diagram

Test condition

Statement 1

Statement 2

Example

if (D > 0)
    printf("x1=%f, x2=%f\n", x1, x2);
else
    printf("No real roots exist\n");
Relational operators

Less: \( < \)
Less or equal: \( \leq \)
Greater: \( > \)
Greater or equal: \( \geq \)
Equal: \( = = \)
Not equal: \( != \)

Example

```c
if (x==0)
    printf("Zero");
else
    printf("Not zero");
```
Cascaded if-else and compound statements

```c
if (D > 0)
{
    x1 = (-b + sqrtf(D)) / (2 * a);
    x2 = (-b - sqrtf(D)) / (2 * a);
} /* Compound statement */
else if (D == 0)
    x1 = -b / (2 * a); /* Cascaded if */
/* Simple statement */
```
Example: solving $ax^2 + bx + c = 0$

```c
/* solution of the quadratic equation $ax^2 + bx + c = 0$ */
#include <stdio.h> /* needed for printf and scanf */
#include <math.h> /* needed for sqrtf */

int main()
{
    float a, b, c; /* quadratic equation coefficients */
    float D; /* determinant */
    float x1, x2; /* solution(s) */

    /* get equation coefficients */
    printf("Enter a, b, and c: ");
    scanf("%f %f %f", &a, &b, &c);
    printf("Solving equation \(ax^2+bx+c=0\)\n", a, b, c);

    /* compute solution */
    D = b * b - 4 * a * c; /* compute determinant */
    if (D > 0) /* two real roots exist */
    {
        x1 = (-b + sqrtf(D)) / (2 * a);
        x2 = (-b - sqrtf(D)) / (2 * a);
    }
    else if (D == 0) /* only one root exists */
        x1 = -b / (2 * a);

    /* print results */
    if (D > 0)
        printf("x1=%.2f, x2=%.2f\n", x1, x2);
    else if (D == 0)
        printf("x=%.2f\n", x1);
    else
        printf("No real roots exist\n");
    return 0;
}
```