CS 3710 Vocational Languages

September 10, 2012

C Statements

- A statement is a part of your program that can be executed.
- A statement is followed by a semicolon.
- A declaration is a statement: `int a;`
- An expression is a statement: `a=a+1;`
- A function call is also a statement: `printf("%d",a);`
- Multiple statements can be grouped using `{` into compound statements. `{ a=a+1; b=b+1; }
- if-statement, switch-statement, loop-statement

The if Statement

- Form 1:
  ```c
  if (expression)
  statement1;
  next statement;
  ```

- Form 2:
  ```c
  if (expression)
  statement1;
  else
  statement2;
  next statement;
  ```

- Form 3:
  ```c
  if (expression1)
  statement1;
  else if (expression2)
  statement2;
  else
  statement3;
  next statement;
  ```

- If the expression is non-zero (i.e., it does not have to be exactly 1)
  execute statement1

- If expression1 is evaluated before expression2
  expression1 and expression2 are mutually exclusive

Exercise

```c
if (amount <= balance)
balance = balance - amount;
else
balance = balance - OVERDRAFT_PENALTY;

if (amount <= balance)
balance = balance - amount;
else
balance = balance - OVERDRAFT_PENALTY;

if (amount > balance)
balance = balance - OVERDRAFT_PENALTY;
if (value >= 0.0)
value *= 2.0;
else if (value >= 5.0)
value *= 2.5;
else
value *= 5.0;
```
The switch statement

- Alternative to if-statement for "multiple–way decision".
- Generic form:

  ```java
  switch(expression) { //x, 1, 3
  case constant_expr1 : statements
  case constant_expr2 : statements
  case constant_exprk : statements
  default : statements
  }
  ```

  1. expression is evaluated to be an integer value;
  2. The program jumps to the corresponding constant_expr (1, 2)
  3. All statements after the constant_expr are executed until a break (or goto, return) statement is encountered.

Example

```java
switch (c) {
  case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8':
    digit(c-'0')++; break;
  case ' ':case '\n': case '\t':
    nwhite++; break;
  default:
    another++; break;
}
```

"break" break out the switch statement

Without "break", all statements followed (nwhite++; break;) in the switch statement will be executed.

switch vs. if-else statement

```c
int a;
\n");
printf("2. Save file.\n");
printf("3. Save as..\n");
printf("4. Quit.\n");
scanf("Your choice: ", &a);
if(a==1)
  open_file();
else if(a==2)
  save_file();
else if(a==3)
  save_as();
else if(a==4) return 0;
else return 1;
```

```c
int a;
\n");
printf("2. Save file.\n");
printf("3. Save as..\n");
scanf("Your choice: ", &a);
switch(a) {
  case 1: open_file();break;
  case 2: save_file();break;
  case 3: save_as();break;
  case 4: return 0;
  default: return 1;
}
```

Loop Structures

```c
initialization;
while (conditions) {
  statements; update;
}
```
- Check condition before executing statements;

```c
initialization;
do { statements; update; } while (conditions)
```
- Execute statements before checking conditions

```c
for(initialization; conditions; update) { statements; }
```
Converting loop structures

for (i=0; i < 3; i++)
    printf("%d", i);

While:  i=0;
    while (k < 3)
        {printf("%d", i);i++;}
Do-while: i=0;
    do(printf("%d", i);i++;)
    while (k < 3)

Converting Nested Loops

for (int i = 0; i < 3 ; i++)
    for (int j = 0; j < 3; j++)
        printf("%d %d\n",i,j);

Common Errors: Semicolons

• A missing semicolon:
  ```c
  for (years = 1;
     (balance = balance + balance * rate / 100) <
     targetBalance;)
  {empty body. add ; at the end of the for statement}
  ```

• A semicolon that shouldn’t be there:
  ```c
  for(i=1;i<=10;i++)
  {double interest = balance * rate / 100;
   balance = balance + interest;
  }
  ```

Exercise

<table>
<thead>
<tr>
<th>Loop</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>int i=5;int sum=0; while(i &lt;=10);</td>
<td></td>
</tr>
<tr>
<td>{sum=sum+i;</td>
<td></td>
</tr>
<tr>
<td>i++;</td>
<td></td>
</tr>
<tr>
<td>printf(&quot;%d \n&quot;,sum); }</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int i=5;</th>
</tr>
</thead>
<tbody>
<tr>
<td>while (i-- &gt; 0);</td>
</tr>
<tr>
<td>{printf(&quot;%d \n&quot;,i); }</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int i; for(i=5; i!=0; i=i-2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>printf(&quot;%d \n&quot;,i);</td>
</tr>
</tbody>
</table>
Trace For-Loop

```java
for (int i=1; i<4; i++)
    for (int j=4; j>0; j--)
        if ((i+j)%2 == 0)
            println("%d", 0);
        else
            println("%d", 1);
```

Trace Do-While Loop

```java
int i=0;
do {
    int j =1;
do {
        println("%c", 'a');
j++;
    } while(j < i);
    println("\n");
i++;
} while (i < 5);
```

Trace While-Loop

```java
int i=10;
while(i > 0)
{
    int j =0;
    while (j < 10)
    {
        println("%d ", j);
j=j+2;
    }
println("\n");
i=i-j;
}```