



Loops II

Chapter 6



The `for` Loop

- A more compact way to express the loop control information.
- All control information in a single line at the top of the loop.

Initialization	Condition	Update
<code>i = 1;</code>	<code>i <= 10;</code>	<code>i++</code>
<code>for (</code>		
<code>{</code>		
<code>sum += i;</code>	Loop body	
<code>}</code>		
<code>)</code>		

- Most widely used looping construct in C



The `for` Loop

Initialization is done only once, before the loop body is executed the first time.

Condition is tested *before* loop body is executed.

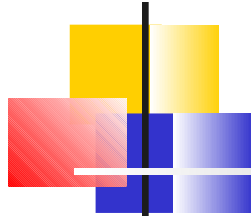
(Same as a “while” loop. Loop body is executed if condition is true)

```
for ( i = 1; i <= 10; i++ )  
{  
    sum += i;  
}
```

Update is done *after* the loop body is executed each time, regardless of the condition.

Repeat the loop body if condition *is* true. (Like “while”.)

Continue after loop body if condition is *not* true.



The `for` Loop

Things to notice

The three sections of the control unit are separated by semicolons.

```
for (i = 1;    i <= 10;    i++)  
{  
    sum += i;  
}
```

↑
No semicolon!

Loop body is a block of code, delimited by curly brackets.

Same as for while or if.

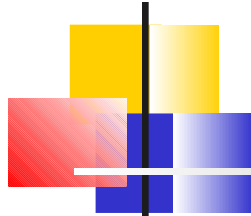


The for Loop

- Caution: Watch out for this mistake:

```
for (i = 1, i <= 10, i++)  
{  
    sum += i;  
}
```

Should be semicolons!



The `for` Loop

Programming Style

```
for (i = 1;    i <= 10;    i++)  
{  
    sum += i;  
}
```

The usual “code block”

Indent the code four spaces.

Align curly brackets with the “for”.



The `for` Loop

Legally you can modify the loop control variable inside the loop

```
for (i = 1;    i <= 10;    i++)  
{  
    i += 3;  
    sum += i;  
}
```

This is always a bad idea!

Treat the control variable as a read-only variable inside the loop.



Example: squares.c

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int i = 0;
```

```
    printf ("This program outputs a table of squares\n");
```

```
    for (i = 0; i <= 10; i++)
```

```
    {
```

```
        printf ("%3d  %6d\n", i, i*i);
```

```
    }
```

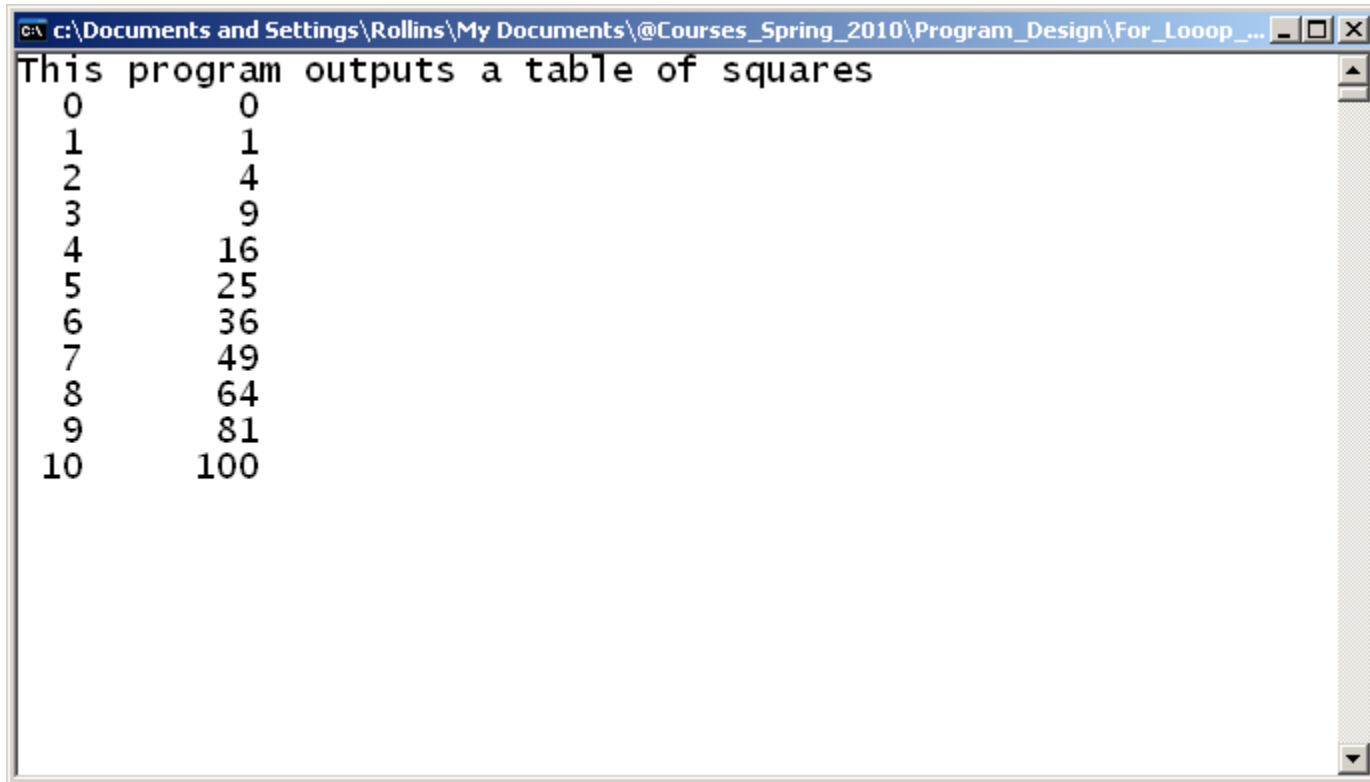
```
    getchar();    // Keep window open
```

```
    getchar();
```

```
    return 0;
```

```
}
```


Program Running



```
c:\Documents and Settings\Rollins\My Documents\@Courses_Spring_2010\Program_Design\F...
This program outputs a table of squares
0      0
1      1
2      4
3      9
4      16
5      25
6      36
7      49
8      64
9      81
10     100
```

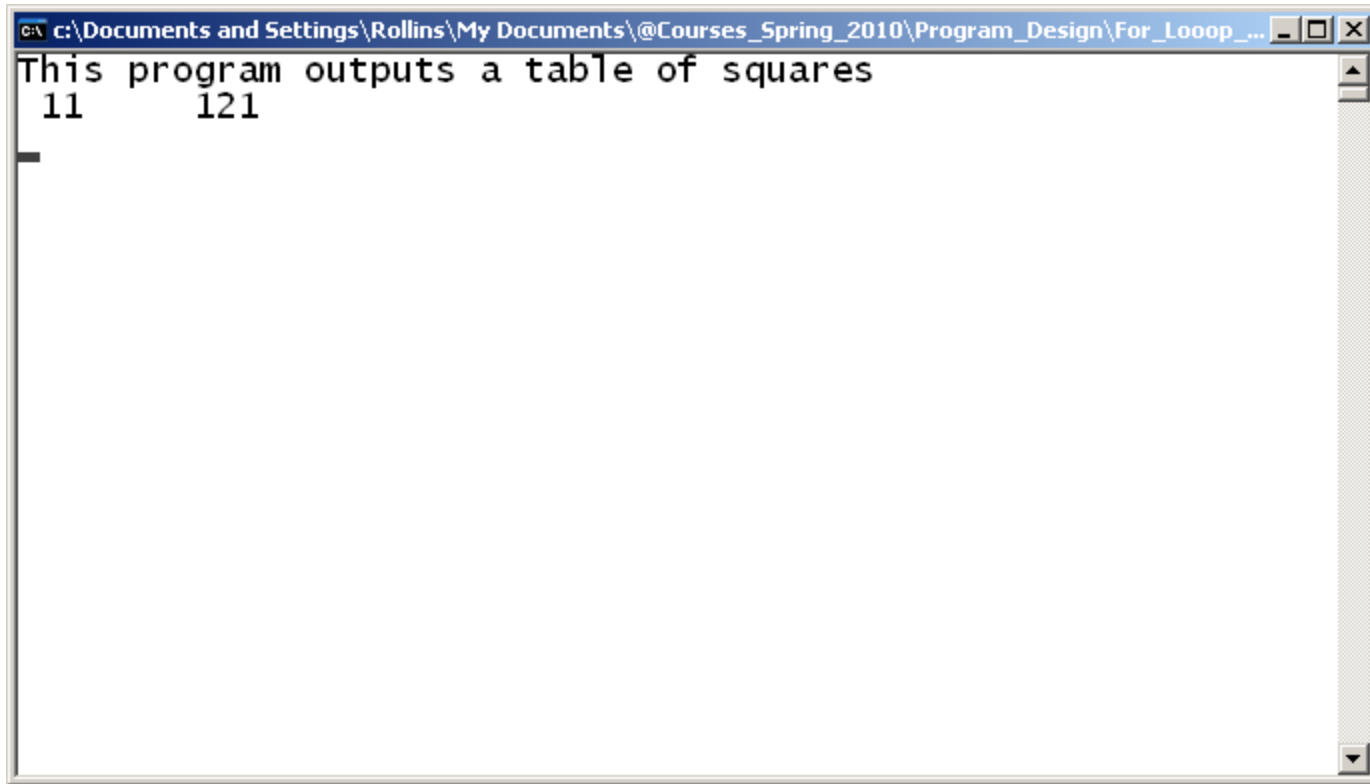


Common Mistakes

- See what happens if you
 - put a semicolon after the control statement

```
for (i = 0; i <= 10; i++);  
{  
    printf ("%3d  %6d\n", i, i*i);  
}
```

Program Running



```
c:\Documents and Settings\Rollins\My Documents\@Courses_Spring_2010\Program_Design\For_Loop_...
This program outputs a table of squares
11      121

```

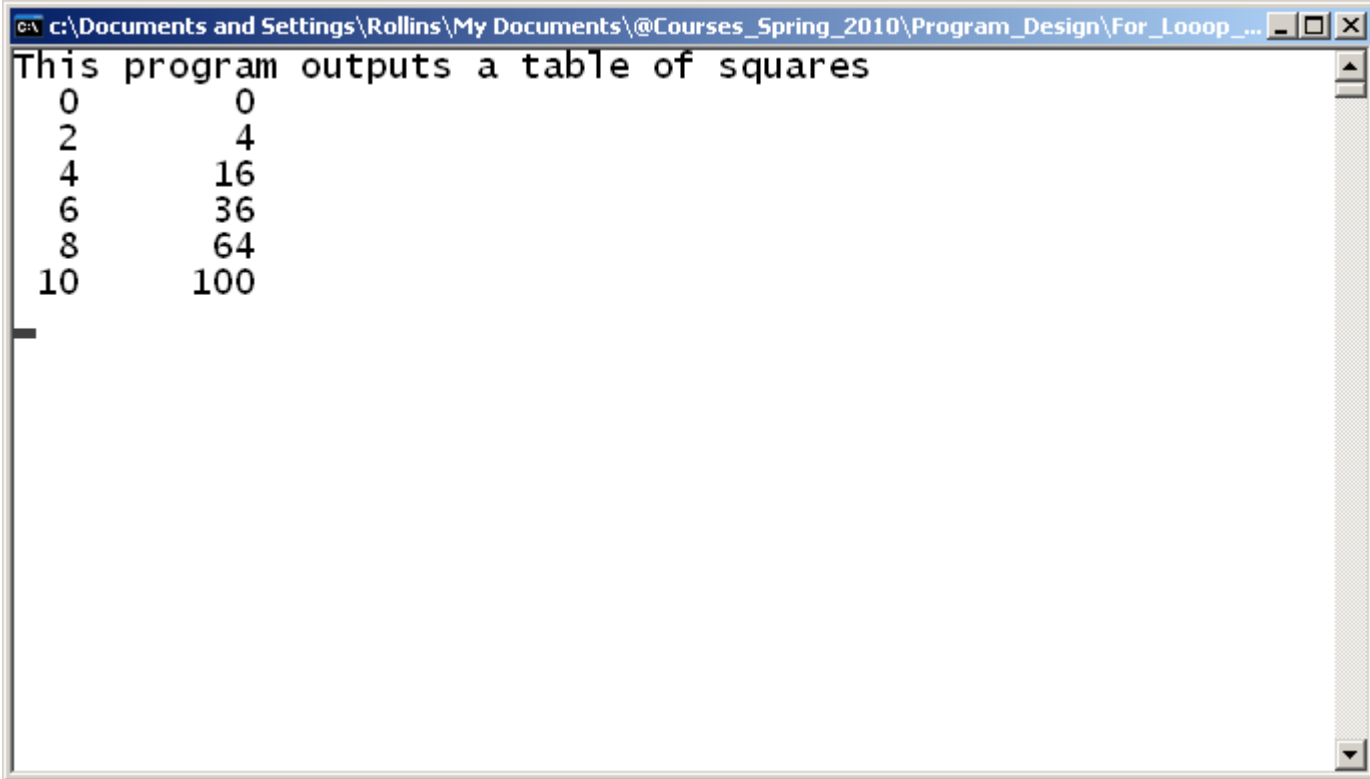


Common Mistakes

- See what happens if you
 - increment the control variable inside the loop body

```
for (i = 0; i <= 10; i++)  
{  
    printf ("%3d  %6d\n", i, i*i);  
    i++;  
}
```

Program Running



```
c:\Documents and Settings\Rollins\My Documents\@Courses_Spring_2010\Program_Design\For_Loop_...
This program outputs a table of squares
 0      0
 2      4
 4     16
 6     36
 8     64
10    100

```

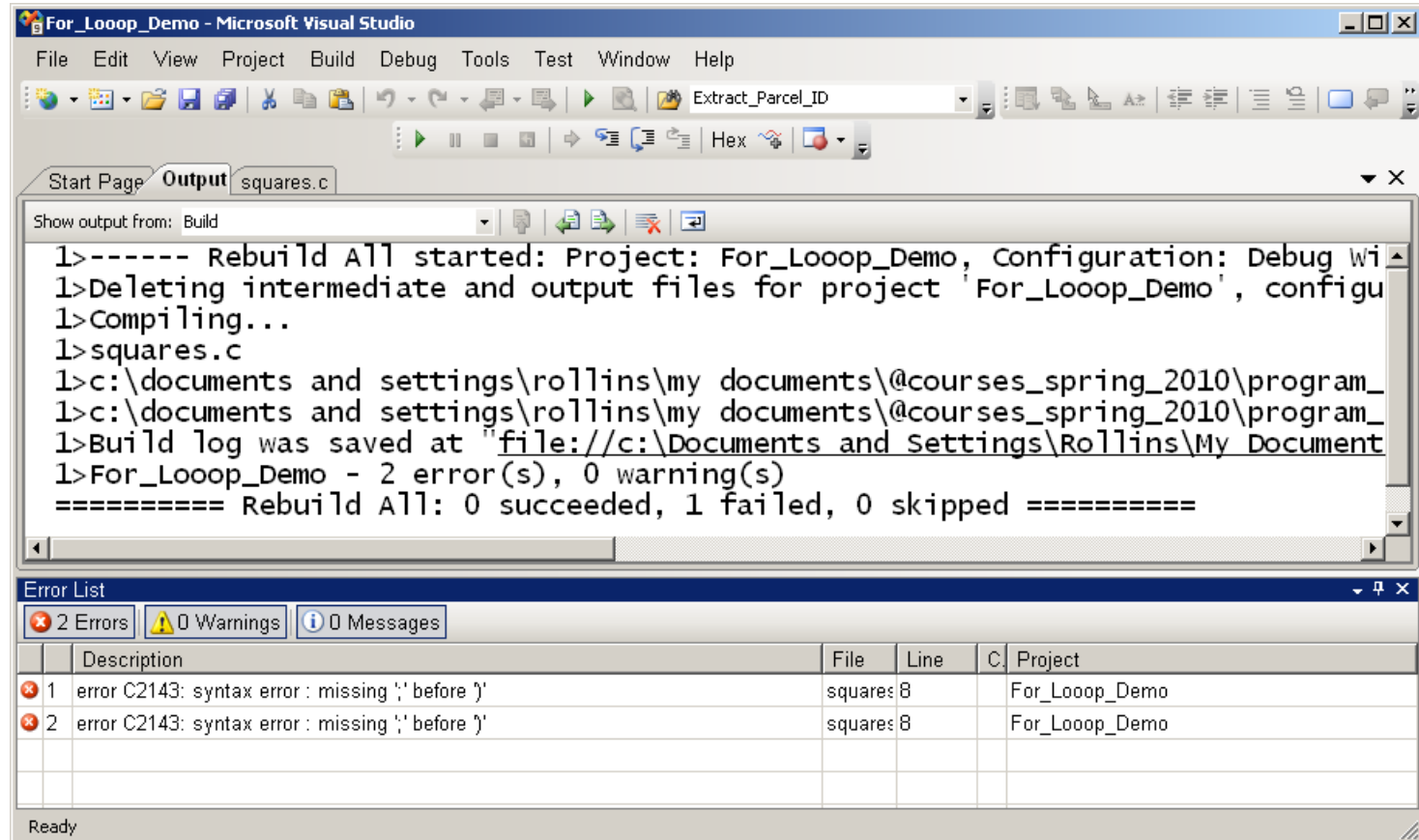


Common Mistakes

- See what happens if you
 - use commas rather than semicolons in the control statement

```
for (i = 1, i <= 10, i++)  
{  
    printf ("%3d  %6d\n", i, i*i);  
    i++;  
}
```

Compile Error





Alternative Test

- You could test for inequality rather than less than.
 - Usually NOT a good idea!

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int i = 0;
```

```
    printf ("This program outputs a table of squares\n");
```

```
    for (i = 0; i != 10; i++)
```

```
    {
```

```
        printf ("%3d  %6d\n", i, i*i);
```

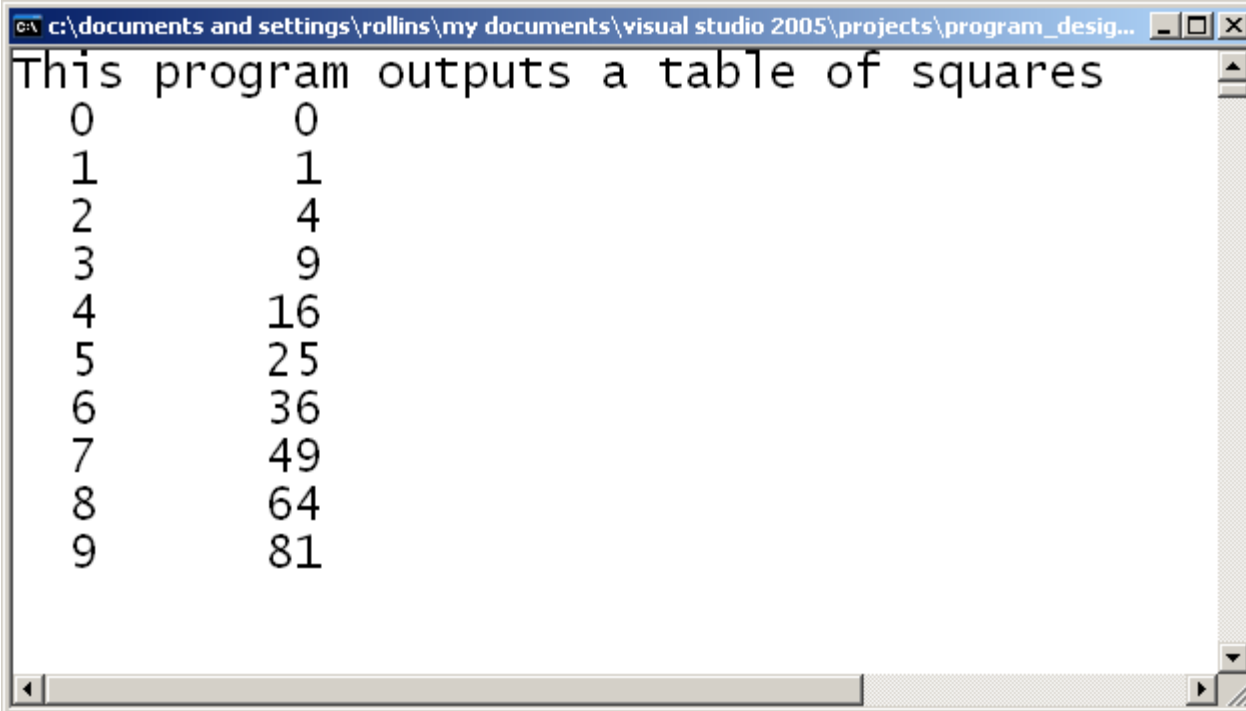
```
    }
```

```
    getchar();    // Keep window open
```

```
    getchar();
```

```
    return 0;
```


Works Fine



```
c:\documents and settings\rollins\my documents\visual studio 2005\projects\program_desig...  
This program outputs a table of squares  
0      0  
1      1  
2      4  
3      9  
4     16  
5     25  
6     36  
7     49  
8     64  
9     81
```

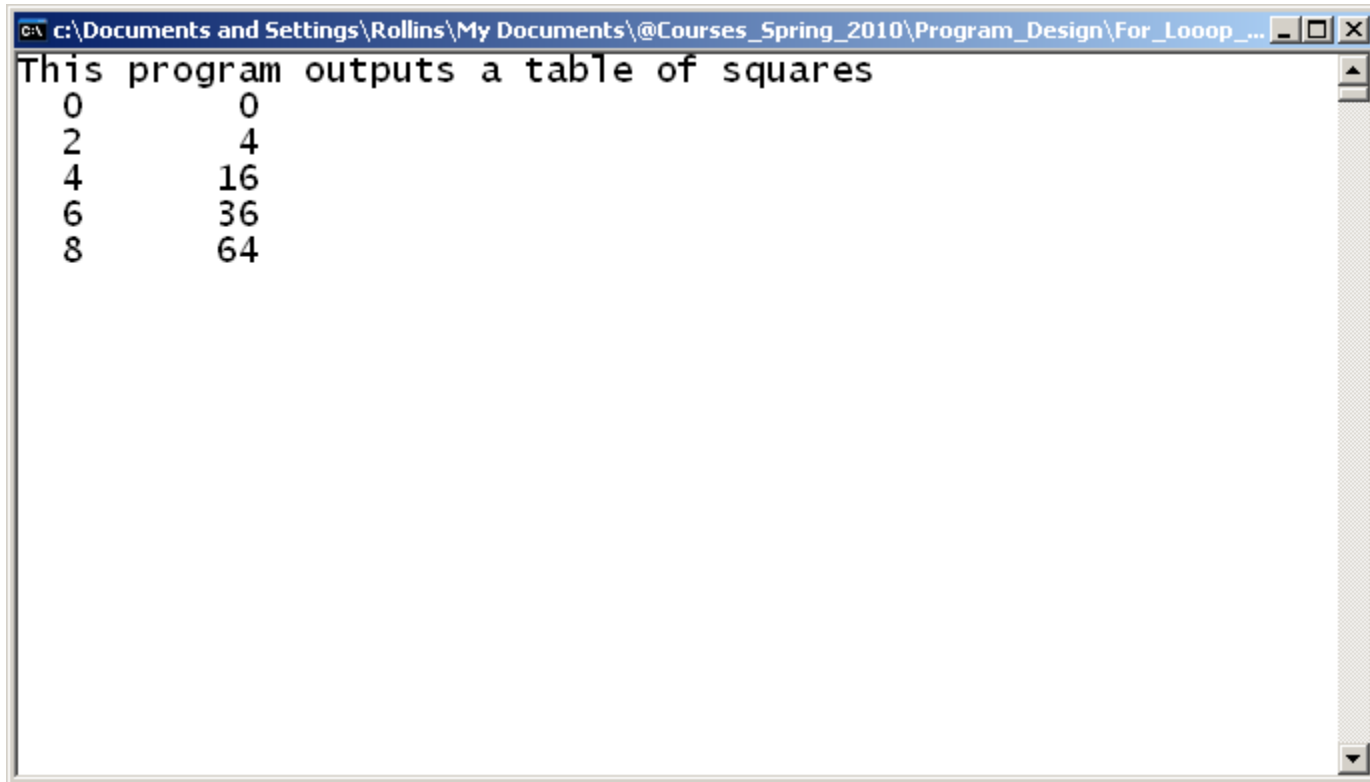


Variations

- What if we want squares of just the even numbers?

```
for (i = 0; i != 10; i+=2)
{
    printf ("%3d  %6d\n", i, i*i);
}
```

Works Fine!



```
c:\Documents and Settings\Rollins\My Documents\@Courses_Spring_2010\Program_Design\For_Loop_...
This program outputs a table of squares
0      0
2      4
4      16
6      36
8      64
```

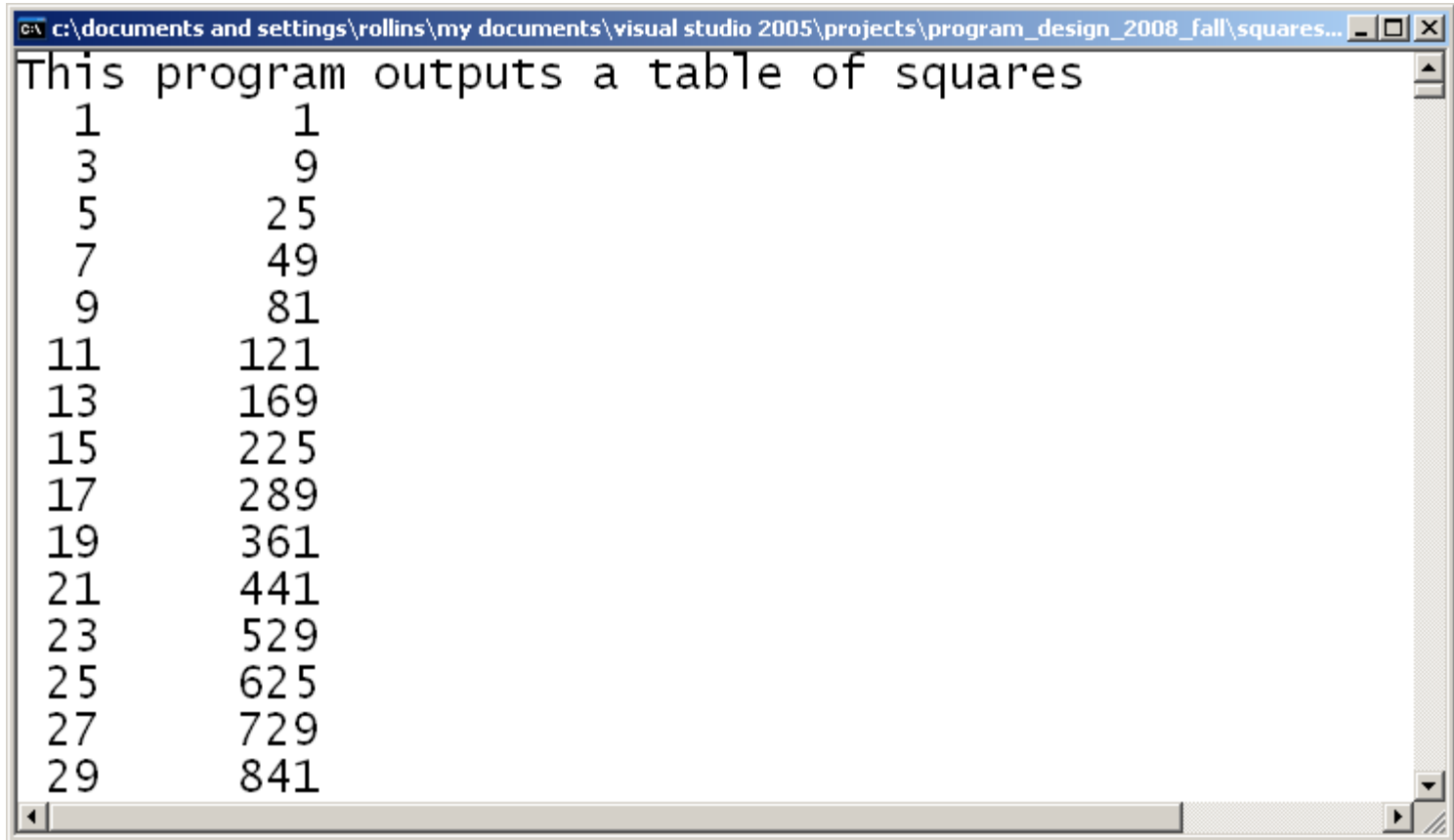


Variations

- How about just the odd numbers?

```
for (i = 1; i != 10; i+=2)
{
    printf ("%3d  %6d\n", i, i*i);
}
```

Not so good!



```
c:\documents and settings\rollins\my documents\visual studio 2005\projects\program_design_2008_fall\squares...
This program outputs a table of squares
 1      1
 3      9
 5     25
 7     49
 9     81
11    121
13    169
15    225
17    289
19    361
21    441
23    529
25    625
27    729
29    841
```

Have to stop with Ctrl-C



Lesson Learned

- On tests for inequality, the limit value must be *exactly right*.
- Prefer tests for *less than* over tests for *not equal* to stop a loop.
 - Be sure the loop stops even if the limiting value is not exactly right!

End of Section



Modifying Control Flow

- C provides several statements that modify the normal flow of control within a loop:
 - break
 - continue
 - goto
- break and continue are OK.
 - Use as appropriate to break out of a loop or immediately start the next iteration.
- goto is bad.
 - A relic of the olden days.
 - Kept mainly for compatibility.
 - Forget that it is there

- C99 permits the loop control variable to be defined in the control statement:

```
for (int i = 0; i <= 10; i++)  
{  
    printf ("%3d  %6d\n", i, i*i);  
}
```

- `i` will not be visible outside the loop.
- This construct was invented in C++
 - Adopted for C in C99



Using C99

- Circe supports C99
 - `gcc -Wall -std=c99 xxx.c`
- but Visual Studio does not.



Compile and Run on Circe

```
turnerr@login2:~/test
[turnerr@login2 test]$
[turnerr@login2 test]$ cat squares.c
#include <stdio.h>

int main()
{
    printf ("This program outputs a table of squares\n");
    for (int i = 1; i < 10; i++)
    {
        printf ("%3d  %6d\n", i, i*i);
    }

    getchar();    // Keep window open
    getchar();
    return 0;
}
[turnerr@login2 test]$ gcc -Wall -std=c99 squares.c
[turnerr@login2 test]$ ./a.out
This program outputs a table of squares
1      1
2      4
3      9
4     16
5     25
6     36
7     49
8     64
9     81
```



Assignment

- Read Chapter 6
 - Including Q & A section.
- If anything doesn't make sense, ask for help.