| The while Looping<br>Structure      |   |
|-------------------------------------|---|
| CMSC 104, Fall 2012<br>John Y. Park |   |
|                                     | 1 |

## The while Looping Structure

#### Topics

- The while Loop
- Program Versatility
  - Sentinel Values and Priming Reads
- Checking User Input Using a while Loop

#### Reading

• Section 3.7

#### **Review: Repetition Structure**

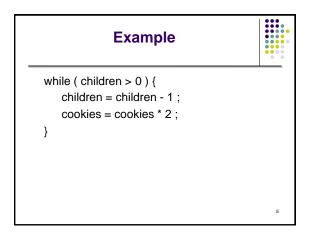


- A **repetition structure** allows the programmer to specify that an action is to be repeated while some condition remains true.
- There are three repetition structures in C, the while loop, the for loop, and the do-while loop.

### The while Repetition Structure

while ( condition ) {
 statement(s)
}

The braces are not required if the loop body contains only a single statement. However, they are a good idea and are required by the 104 C Coding Standards.



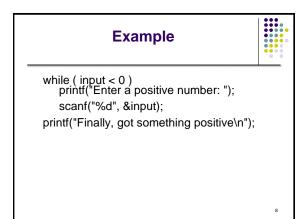


- Always place braces around the body of a while loop.
- Advantages:
  - Easier to read
  - Will not forget to add the braces if you go back and add a second statement to the loop body
  - Less likely to make a semantic error
- Indent the body of a while loop 3 to 4 spaces
   -- be consistent!





while ( input < 0 )
 scanf("%d", &input);
printf("Finally, got something positive\n");</pre>



# Another while Loop Example



- <u>Problem</u>: Write a program that calculates the average exam grade for a class of 10 students.
- What are the program inputs?
  - the exam grades
- What are the program outputs?
  - the average exam grade

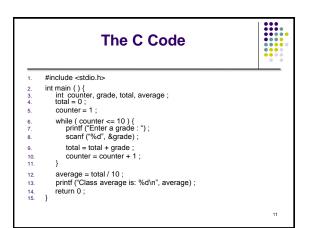
# The Pseudocode

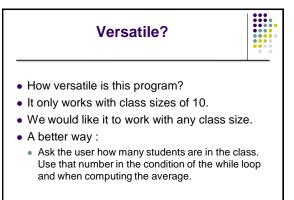


10

<total> = 0 <grade\_counter> = 1 While (<grade\_counter> <= 10) Display "Enter a grade: "

Read <grade> <total> = <total> + <grade> <grade\_counter> = <grade\_counter> + 1 End\_while <average> = <total> / 10 Display "Class average is: ", <average>



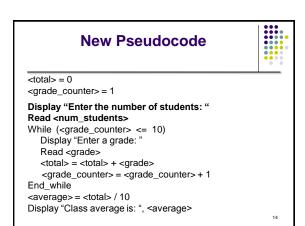


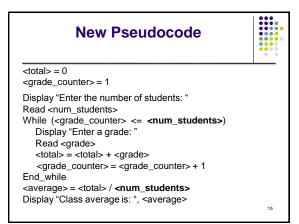
# New Pseudocode

13

<total> = 0 <grade\_counter> = 1

While (<grade\_counter> <= 10) Display "Enter a grade: " Read <grade> <total> = <total> + <grade> <grade\_counter> = <grade\_counter> + 1 End\_while <average> = <total> / 10 Display "Class average is: ", <average>





|                             | New C Code  |    |
|-----------------------------|---|----|
| 1.<br>2.<br>3.              | <pre>#include <stdio.h> int main () {     int <b>mumStudents</b>, counter, grade, total, average;</stdio.h></pre>   |    |
| 4.<br>5.                    | total = 0 ;<br>counter = 1 ;  |    |
| 6.<br><b>7.</b><br>8.<br>9. | <pre>printf ("Enter the number of students: ");<br/>scanf ("%d", &amp;numStudents);<br/>while ( counter &lt;= numStudents) {<br/>printf ("Enter a grade : ");<br/>scanf ("%d", &amp;grade);</pre> |    |
| 11.<br>12.<br>13.           | <pre>total = total + grade ; counter = counter + 1 ; }</pre>  |    |
| 14.<br>15.<br>16.<br>17.    | verage = total / numStudents ;<br>printf ("Class average is: %d\n", average) ;<br>return 0 ;  | 16 |

#### Why Bother to Make It Easier?



- Why do we write programs?So the user can perform some task
- The more versatile the program, the more difficult it is to write. BUT it is more useable.
- The more complex the task, the more difficult it is to write. But that is often what a user needs.
- Always consider the user first.

#### **Using a Sentinel Value**



- We could let the user keep entering grades and when he's done enter some special value that signals us that he's done.
- This special signal value is called a **sentinel value**.
- We have to make sure that the value we choose as the sentinel isn't a legal value. For example, we can't use 0 as the sentinel in our example as it is a legal value for an exam score.

# **The Priming Read**

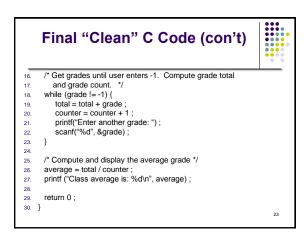


- When we use a sentinel value to control a while loop, we have to get the first value from the user before we encounter the loop so that it will be tested and the loop can be entered.
- This is known as a priming read.
- We have to give significant thought to the initialization of variables, the sentinel value, and getting into the loop.

| New Pseudocode  |    |
|---|----|
| <total> = 0<br/><grade_counter> = 0</grade_counter></total>   |    |
| Display "Enter a grade: "<br>Read <grade><br/>While (<grade> != -1)<br/><total> = <total> + <grade><br/><grade_counter> = <grade_counter> + 1<br/>Display "Enter another grade: "</grade_counter></grade_counter></grade></total></total></grade></grade> |    |
| Read <grade><br/>End_while<br/><average> = <total> / <grade_counter><br/>Display "Class average is: ", <average></average></grade_counter></total></average></grade>  |    |
| Display Class average is: , saverage  | 20 |

|            | New C Code   |    |
|------------|--|----|
| 1.         | #include <stdio.h></stdio.h>                         |    |
| 2.<br>3.   | int main () {<br>int counter, grade, total, average; |    |
| 4.         | total = 0 ;  |    |
| 5.         | counter = 0;   |    |
| 6.<br>7.   | printf("Enter a grade: ") ;<br>scanf("%d", &grade) ; |    |
| 7.<br>8.   | while (grade != -1) {                                |    |
| 9.         | total = total + grade ;                              |    |
| 10.        | counter = counter + 1 ;                              |    |
| 11.        | printf("Enter another grade: ") ;                    |    |
| 12.        | scanf("%d", &grade) ;                                |    |
| 13.        | }  |    |
| 14.        | average = total / counter ;                          |    |
| 15.        | printf ("Class average is: %d\n", average) ;         |    |
| 16.<br>17. | return 0 ;   | 21 |

|          | Final "Clean" C Code  |    |  |
|----------|---|----|--|
| 1.       | #include <stdio.h></stdio.h>  |    |  |
| 2.<br>3. | int main () {   |    |  |
| ŀ.       | int counter ; /* counts number of grades entered */                       |    |  |
| 5.       | int grade ; /* individual grade */  |    |  |
| i.       | int total; /* total of all grades */<br>int average ; /* average grade */ |    |  |
| i.       | In average, / average grade /   |    |  |
|          | /* Initializations */   |    |  |
| 10.      | total = 0;  |    |  |
| 11.      | counter = 0;  |    |  |
| 12.      |   |    |  |
| 13.      | /* Priming read to get initial grade from user */                         |    |  |
| 14.      | printf("Enter a grade: ");  |    |  |
| 15.      | scanf("%d", &grade);  | 22 |  |
|          |   |    |  |



| U         | sing a while Loop to Check User<br>Input                           |    |
|-----------|--|----|
| 1.        | #include <stdio.h></stdio.h>                                       |    |
| 2.        | int main () {  |    |
| 3.        | int number ;   |    |
| 4.        | printf ("Enter a positive integer : ");                            |    |
| 5.        | scanf ("%d", &number) ;  |    |
| 6.        | while ( number <= 0 ) {  |    |
| 7.        | printf ("\nThat's incorrect. Try again.\n");                       |    |
| 8.<br>9   | printf ("Enter a positive integer: ") ;<br>scanf ("%d", &number) ; |    |
| 3.<br>10. | }  |    |
| 11.       | printf ("You entered: %d\n", number) ;                             |    |
| 12.       | return 0;  |    |
| 13.       | }  |    |
|           |  |    |
|           |  | 24 |