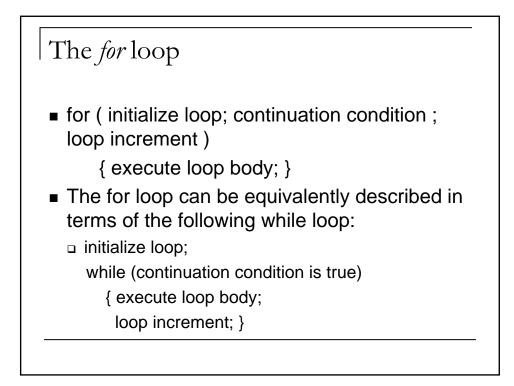
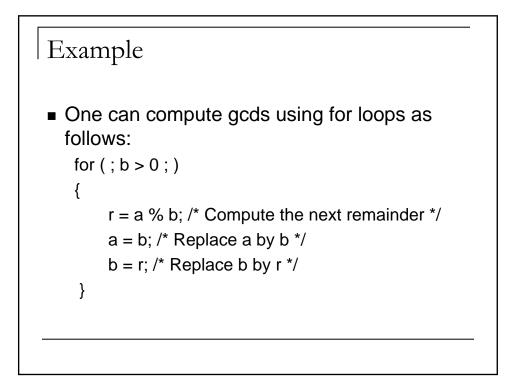
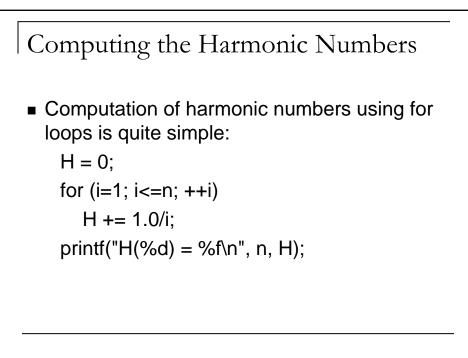
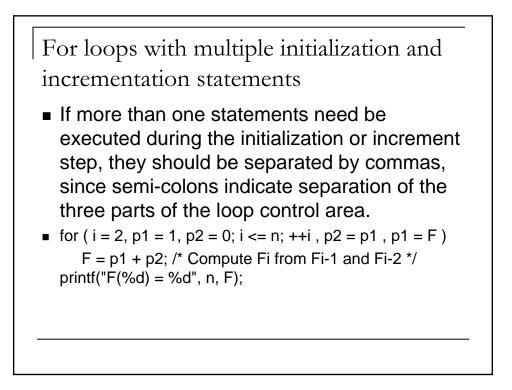


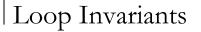
Loops



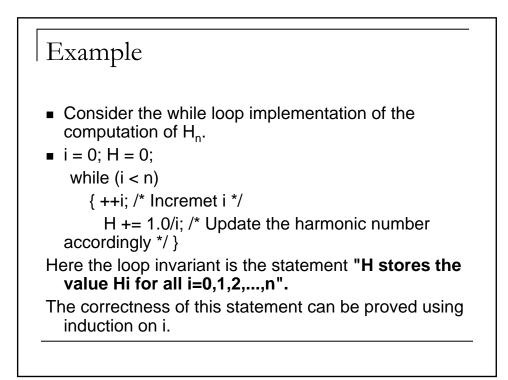






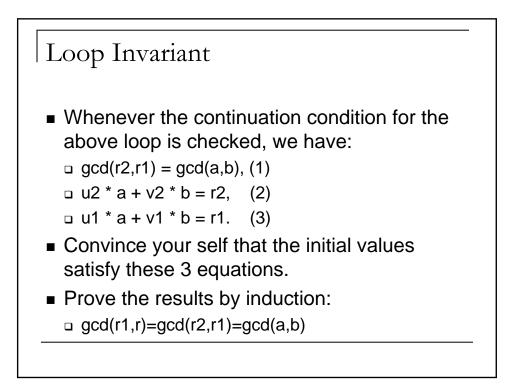


- For verifying the correctness of loops one often uses the concept of loop invariance.
- A loop invariant refers to a statement that is true at all instants when the loop condition is checked.
- It may be expressed in terms of one or more variables controlling the flow of the loop.



## Another example

/\* Initialize \*/
r2 = a; u2 = 1; v2 = 0; /\* Previous-to-previous values \*/
r1 = b; u1 = 0; v1 = 1; /\* Previous values \*/
/\* Extended gcd loop \*/
while (r1 > 0) {
 /\* Compute values for the current iteration \*/
 q = r2 / r1; /\* Compute the next quotient \*/
 r = r2 - q \* r1; /\* Compute the next remainder \*/
 u = u2 - q \* u1; /\* Identically compute the next u value \*/
 v = v2 - q \* v1; /\* Identically compute the next v value \*/
 r = r2 = r1; u2 = u1; v2 = v1; /\* Let the previous-to-previous values be the
 previous values \*/
r1 = r; u1 = u; v2 = v; /\* Let the previous values be the current values \*/ }
 printf("gcd(a,b) = %d = (%d) \* a + (%d) \* b\n", r2, u2, v2);



for a sample run with a=78 and b=21.											
teration No	r2	r1	u2	u1	v2	v1	q	r	u	v	u2*a+v2*b
Before loop	78	21	1	0	0	1	-	-	-	-	78
1	78 21	21 15	1 0	0 1	0 1	1 -3	3 3	15 15	1 1	-3 -3	78 21
2	21 15	15 6	0 1	1 -1	1 -3	3 4	1 1	6 6	-1 -1	4	21 15
3	15 6	6 3	1 -1	-1 3	-3 4	4 -11	2 2	3 3	3 3	-11 -11	15 6
4	6 3	3 0	-1 3	3 -7	4 -11	-11 26	2 2	0 0	-7 -7	26 26	6 3

