

for $i = 0$ to 10 step 2

for $i = 10$ ^{down to} ~~to~~ 0 step -2

for $i = \text{start}$ to end step s

if ($s > 0$)

{ for ($i = \text{start}; i \leq \text{end}; i += s$)
 ;
 }

} else { for ($i = \text{start}; i \geq \text{end}; i += s$)
 ;
 }

for ($\text{curr} = \text{start}; \text{curr} \neq \text{NULL}; \text{curr} = \text{curr} \rightarrow \text{next}$) }

for ($\text{int } i = 0; i \leq 10; i += 2$)
 for ($\text{int } i = 10; i \geq 0; i -= 2$)

for ($\text{int } i = 0; i < n; i++$)
 { for ($\text{int } j = 0; j < i; j++$)
 ;
 ;
 ;
 }

avoid if language forbids
 changing loop index in body
 of loop

} printf ("%d", i)

Java: scope error
 C: outputs n

Pascal: error
 others: n-1

Iterators

Java: objects (hasNext, next)

```

Iterator i = new IntIterator(10);
while (i.hasNext())
    Sys.o.pl(i.next());

public class IntIterator implements Iterator
{
    int val;
    public IntIterator(int start) { val = start; }
    public boolean hasNext() { return true; }
    public Object next() { int old = val;
                          val++;
                          return old;
                        }
}

```

alternative iterator as code :

```

iterator i = intIterator(start)
while (true)
    print i.next()

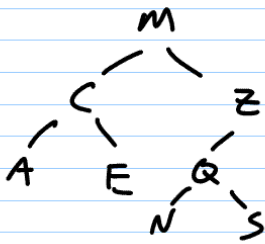
```

```

iterator intIterator(int start)
int val = start
yield (start)
while (true) {
    val++
    yield (val)
}

```

Depth First Iterator



Java: ugh!

need to simulate recursion so
we can do one step per next

alternative:

```

inorder(node)
if (node == null)
  return
else
  inorder(left)
  yield (node.val)
  inorder(right)
  
```

2nd alt: iterator as thread

```

while ( (in = readline()) != null )
{
  ...
}

```

(in = readline()) != null && in.length() > 0

⇒

```

in = readline()
while ( in != null )
{
  ...
  in = readline()
}

```

in != null && in.length() > 0

```

loop
{
  when cond exit
  ...
  when cond exit
}
end

```