

Elements of Algorithms

Mark Maloof
Department of Computer Science
Washington, DC 20057

September 23, 2014

Four Critical Elements

- ▶ Algorithms have some subset of the following critical elements:
 1. simple statements, including but not limited to:
 - ▶ input statements
 - ▶ output statements
 - ▶ assignment statements
 2. sequences of statements, which are also statements
 3. branching statements
 4. looping statements

Algorithm for Simple Interest

- 1: **input** r , b , and m
- 2: $i \leftarrow r \cdot b \cdot m$
- 3: **output** i

- ▷ input statement
- ▷ assignment statement
- ▷ output statement

Another Algorithm for Simple Interest

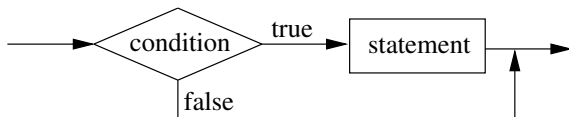
1: **input** r
2: **input** b
3: **input** m
4: $i \leftarrow r$
5: $i \leftarrow i \cdot b$
6: $i \leftarrow i \cdot m$
7: **output** i

▷ input statement
▷ input statement
▷ input statement
▷ assignment statement
▷ assignment statement
▷ assignment statement
▷ output statement

Branching: If-then Statement

```
if some condition is true then  
    statement (or sequence)  
end if
```

Flowchart for an if-then Statement



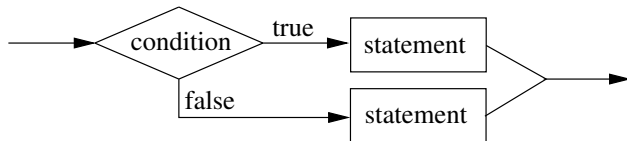
Example of an if-then Statement

```
1: input grade
2: if grade > 64 then
3:   output pass
4: end if
5: if grade ≤ 64 then
6:   output fail
7: end if
```

Branching: If-then-else Statement

```
if some condition is true then  
    statement (or sequence)  
else  
    statement (or sequence)  
end if
```


Flowchart for an if-then-else Statement



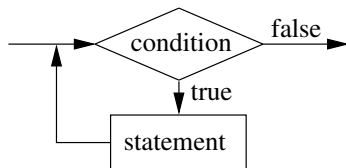
Example of an if-then-else Statement

```
1: input grade
2: if grade > 64 then
3:   output pass
4: else
5:   output fail
6: end if
```

Looping: While Statement, While Loop

```
while some condition is true do  
    statement (or sequence)  
end while
```

Flowchart for a While Loop



Example of a While Loop

```
1: input grade
2: while there is a grade do
3:   if grade > 64 then
4:     output pass
5:   else
6:     output fail
7:   end if
8:   input grade
9: end while
```

Repeat-until Loop

repeat

statement (or sequence)

until some condition is true

Equivalent to:

statement (or sequence)

while some condition is false **do**

statement (or sequence)

end while

For Loop

```
for  $i \leftarrow b, e$  do  
    statement (or sequence)  
end for
```

Equivalent to:

```
 $i \leftarrow b$   
while  $i \leq e$  do  
    statement (or sequence)  
     $i \leftarrow i + 1$   
end while
```

For-each Loop

for each element of some collection **do**
 statement (or sequence)
end for

Equivalent to:

$i \leftarrow 1$
 $e \leftarrow$ the number of elements in the collection
while $i \leq e$ **do**
 element \leftarrow *i*th element of the collection
 statement (or sequence)
 $i \leftarrow i + 1$
end while

Example of a For-each Loop

```
1: Let Grades be a sequence or list of grades
2: input Grades
3: for each grade in Grades do
4:   if grade > 64 then
5:     output pass
6:   else
7:     output fail
8:   end if
9: end for
```

Algorithm for Binary-to-Decimal Conversion

- 1: Let D be a decimal number, set to zero
- 2: Let B be a binary number, set to zero
- 3: **input** B
- 4: Let B' be B with its digits reversed
- 5: $i \leftarrow 0$
- 6: **for each** binary digit $b \in B'$ **do**
- 7: $D \leftarrow D + b \cdot 2^i$
- 8: $i \leftarrow i + 1$
- 9: **end for**
- 10: **output** D

Algorithm for Decimal-to-Binary Conversion

- 1: Let B be an empty sequence of binary digits
- 2: Let D be a decimal number, set to zero
- 3: **input** D
- 4: **while** $D \neq 0$ **do**
- 5: $r \leftarrow D \bmod 2$
- 6: Add r as the left-most digit of B
- 7: $D \leftarrow D \div 2$ (integer division)
- 8: **end while**
- 9: **output** B