Developing Loops from Invariants
Developing a Loop on a Range of Integers

• Given a range of integers $a..b$ to process.

• Possible alternatives
  ▪ Could use a for-loop: for $x$ in range($a,b+1$):
  ▪ Or could use a while-loop: $x = a; \text{ while } x \leq b$:
  ▪ Which one you can use will be specified

• But does not remove the need for invariants
  ▪ **Invariants**: assertion supposed to be true before and after each iteration of the loop
Suppose you are trying to implement the command

Process a..b

Write the command as a postcondition:

post: a..b has been processed.
Developing an Integer Loop (b)

Set-up using while:

```python
while k <= b:
    # Process k
    k = k + 1
# post: a..b has been processed.
```
Developing an Integer Loop (c)

Add the invariant:

```plaintext
# invariant: a..k-1 has been processed

while k <= b:
    # Process k
    k = k + 1

# post: a..b has been processed.
```

Note it is post condition with the loop variable.
Developing an Integer Loop (d)

Fix the initialization:

Init to make invariant true

# invariant: a..k-1 has been processed

while k <= b:
    # Process k
    k = k + 1

# post: a..b has been processed.
Developing an Integer Loop (e)

Figure out how to “Process k”:

init to make invariant true

# invariant: a..k-1 has been processed

while k <= b:

    # Process k

    implementation of “Process k”

    k = k + 1

# post: a..b has been processed.
Range

- Pay attention to range:
  a..b or a+1..b or a...b-1 or ...

- This affects the loop condition!
  - Range a..b-1, has condition k < b
  - Range a..b, has condition k <= b

- Note that a..a-1 denotes an empty range
  - There are no values in it

- a..b how many elements?  \( b - a + 1 \)
Example of an assertion about an sequence \( b \). It asserts that:

1. \( b[0..k-1] \) is sorted (i.e. its values are in ascending order)
2. Everything in \( b[0..k-1] \) is \( \leq \) everything in \( b[k..\text{len}(b)-1] \)
Algorithm Inputs

- We may specify that the list in the algorithm is
  - \(b[0..\text{len}(b)-1]\) or
  - a segment \(b[h..k]\) or
  - a segment \(b[m..n-1]\)

- **Work with whatever is given!**

- Remember formula for # of values in an array segment
  - **Following – First**
  - e.g. the number of values in \(b[h..k]\) is \(k+1-h\).
**Example Question, Fall 2013 Final**

<table>
<thead>
<tr>
<th>pre:</th>
<th>b</th>
<th>h</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td>sorted</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>post:</th>
<th>b</th>
<th>h</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unchanged, values in b[h+1..k]</td>
<td></td>
<td>b[0..k] w/o duplicates</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>inv:</th>
<th>b</th>
<th>h</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>p</td>
<td></td>
<td></td>
</tr>
<tr>
<td>???</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unchanged, values all in b[h+1..k]</td>
<td>b[p+1..k] w/o duplicates</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Example:**
  - Input  [1, 2, 2, 2, 4, 4, 4]
  - Output [1, 2, 2, 2, 1, 2, 4]
Solution to Fall 2013 Final

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>p</th>
<th>h</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>inv:</td>
<td>b</td>
<td>unchanged</td>
<td>Unchanged, values all in b[h+1..k]</td>
<td>b[p+1..k] w/o duplicates</td>
</tr>
</tbody>
</table>

# Assume 0 <= k, so the list segment has at least one element

p = 

h =

# inv: b[h+1..k] is original b[p+1..k] with no duplicates
# b[p+1..h] is unchanged from original list w/ values in b[h+1..k]
# b[0..p] is unchanged from original list

while :
# Assume 0 <= k, so the list segment has at least one element

\[ p = k - 1 \]

\[ h = k - 1 \]

# inv: b[h+1..k] is original b[p+1..k] with no duplicates

# b[p+1..h] is unchanged from original list w/ values in b[h+1..k]

# b[0..p] is unchanged from original list

**while** :
Solution to Fall 2013 Final

<table>
<thead>
<tr>
<th>0</th>
<th>p</th>
<th>h</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>inv:</td>
<td>b unchanged</td>
<td>Unchanged, values all in b[h+1..k]</td>
<td>b[p+1..k] w/o duplicates</td>
</tr>
</tbody>
</table>

# Assume 0 <= k, so the list segment has at least one element

p = k-1
h = k-1

# inv: b[h+1..k] is original b[p+1..k] with no duplicates
# b[p+1..h] is unchanged from original list w/ values in b[h+1..k]
# b[0..p] is unchanged from original list

while 0 <= p:
# Assume 0 <= k, so the list segment has at least one element

\[ p = k - 1 \]
\[ h = k - 1 \]

# inv: \( b[h+1..k] \) is original \( b[p+1..k] \) with no duplicates

# inv: \( b[h+1..k] \) is unchanged from original list w/ values in \( b[h+1..k] \)

# b[0..p] is unchanged from original list

```python
while 0 <= p:
    if b[p] != b[p+1]:
        b[h] = b[p]
        h = h - 1
        p = p - 1
```
DOs and DON’Ts #1

• **DO** use variables given in the **invariant**.
• **DON’T** use other variables.

```plaintext
# invariant: b[h..] contains the sum of c[h..] and d[k..],
# except that the carry into position k-1 is in 'carry'
while _____________ :
    # Okay to use b, c, d, h, k, and carry
    # Anything else should be ‘local’ to while
```
DO double check corner cases!

- \( h = \text{len}(c) \)
- \( \text{while } h > 0: \)
  - What will happen when \( h=1 \) and \( h=\text{len}(c) \)?
  - If you use \( h \) in \( c \) (e.g. \( c[h] \)) can you possibly get an error?

```python
# invariant: b[h..] contains the sum of c[h..] and d[k..],
# except that the carry into position k-1 is in 'carry'
while h > 0:
    ...
```

Range is off by 1. How do you know?
DOs and DON’Ts #3

• DON’T put variables directly above vertical line.

<table>
<thead>
<tr>
<th>h</th>
<th>i</th>
<th>j</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>&lt;= x</td>
<td>x</td>
<td>?</td>
</tr>
</tbody>
</table>

- Where is j?
- Is it unknown or >= x?
Dutch National Flag

- Sequence of 0..n-1 of red, white, blue colors
  Arrange to put reds first, then whites, then blues
- Input is the list b of integers
- Modifies the list according to the invariant.

<table>
<thead>
<tr>
<th>pre: b</th>
<th>post: b</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 0</td>
<td>== 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Inv: b</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 0</td>
</tr>
</tbody>
</table>
def dutch_national_flag(b):
    j = 0; k = 0; m = len(b)
    while k < m:
        if b[k] == 0:
            k = k + 1
        elif b[k] > 0:
            _swap(b, k, m-1)
            m = m - 1
        else: # b[k] < 0
            _swap(b, k, j)
            k = k + 1
        j = j + 1

Inv: b < 0 == 0 ??? > 0

Dutch National Flag
def dutch_national_flag(b):
    j = 0; k = 0; m = len(b)
    while k < m:
        if b[k] == 0:
            k = k + 1
        elif b[k] > 0:
            _swap(b, k, m - 1)
            m = m - 1
        else:  # b[k] < 0
            _swap(b, k, j)
            k = k + 1
        j = j + 1

dutch_national_flag([-3,-1,5,-2,0])

Inv: b < 0 == 0 ??? > 0
len(b)

\[
\begin{array}{|c|c|c|c|c|}
\hline
0 & j & k & m & \text{len}(b) \\
\hline
\end{array}
\]
Questions?