# CIS 5000: Software Foundations Quiz 6 October 23, 2025

Name (printed): _____ Penn ID (8-digit #): _____

Please write your name as it is displayed on Gradescope. Indicate your answers with a cross mark ☒. To correct a mistake, fill in the incorrect box (like this: ■) and make a cross in the correct box.

1 Recall the notion of *program equivalence*, `cequiv`, from the Equiv chapter: Two `Imp` commands `c1` and `c2` are equivalent if for every starting state `st`, either `c1` and `c2` both diverge or both terminate in the same final state `st'`.

For the following pairs of `Imp` programs mark "Equivalent" if the pair of `Imp` programs are equivalent. If they're not equivalent, mark "Inequivalent" and describe a starting state `st` that exhibits the difference.

(a)
```
X := Z ; Y := Z
```
```
Y := Z ; X := Z
```

☒ Equivalent

☐ Inequivalent when `X =` _____, `Y =` _____, `Z =` _____

(b)
```
skip
```
```
while true do skip end
```

☐ Equivalent

☒ Inequivalent in any starting state

(c)
```
while X < Y do
  Y := Y + 1;
  X := X - 1;
end
```
```
while true do skip end
```

☐ Equivalent

☒ Inequivalent when `X >= Y`

(d)
```
while Y < X do
  Y := Y + 1
end
```
```
if Y < X then
  Y := X
else
  skip
end
```

☒ Equivalent

☐ Inequivalent when `X =` _____, `Y =` _____

(e)
```
if X > 0 then
  while X <> 0 do
    X := X - 1
    Y := Y - 1
  end
else skip end
```

```
Y := Y - X;
X := 0
```

☒ Equivalent

☐ Inequivalent when `X` = _____, `Y` = _____

2 Now suppose we extend Imp with the `havoc` command from the `Imp` homework. Recall that `havoc x` assigns the variable x a natural number value *nondeterministically*. To achieve this, we add the following big-step rule for `havoc`:

```
-------------------------------- (E_Havoc)
st =[ havoc x ]=> (x !-> n ; st)
```

For the following pairs of Imp+`havoc` programs, mark "Equivalent" if they are equivalent. If they're not equivalent, mark "Inequivalent" and describe a starting state `st` that exhibits the difference.

(a)
```
if X = 0 then
  havoc X
else
  skip
end
```

```
while X = 0 do
  havoc X
end
```

☐ Equivalent

☒ Inequivalent when `X = 0`

(b)
```
while X > 0 do
  havoc X
end
```

```
X := 0
```

☒ Equivalent

☐ Inequivalent when `X` = _____