COMP 110

CL03

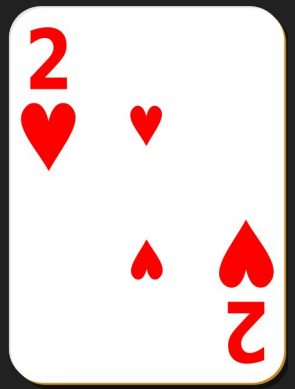
Conditionals

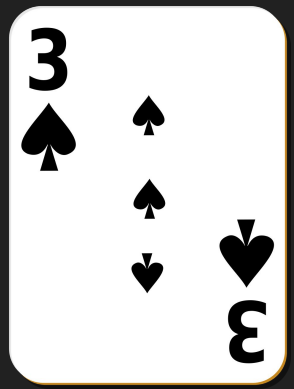
Control Flow

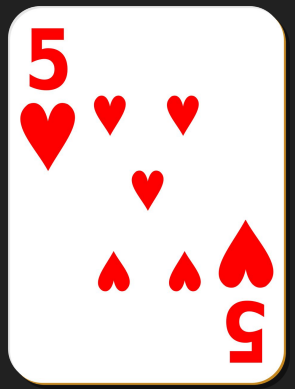
The control flow in a program is the logic which governs what statement the computer will evaluate next, in other words the order in which statements are evaluated.

Python works linearly from top to bottom.

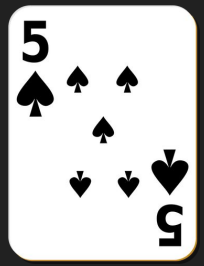
Recall: Finding the Lowest Card





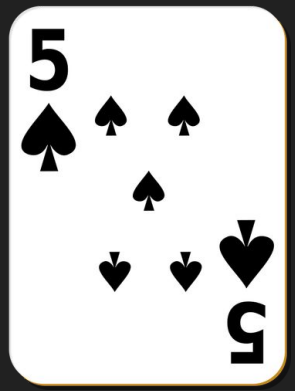


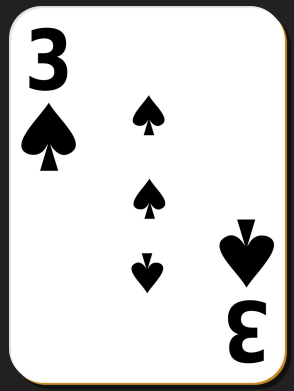
Low card:

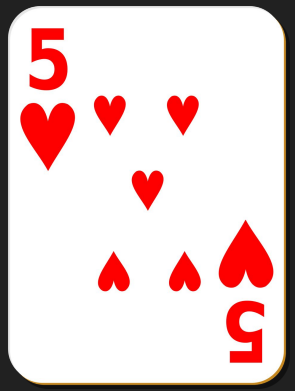


If current card < low card, make it the low card.

Recall: Finding the Lowest Card

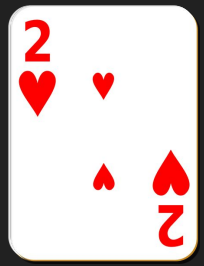






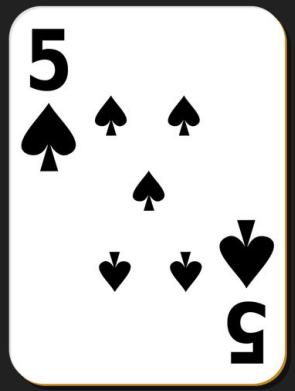
2 < 5? ✓

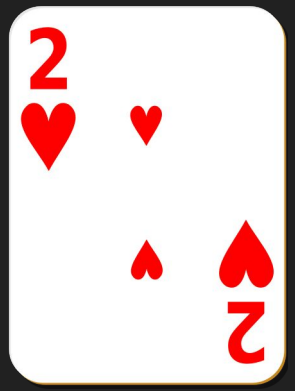
Low card:

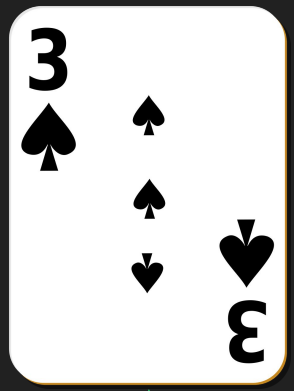


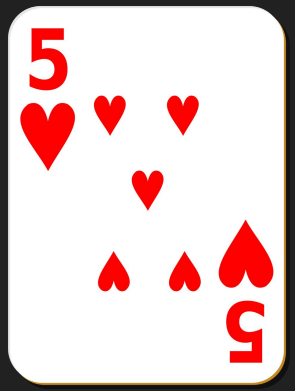
If current card < low card, make it the low card.

Recall: Finding the Lowest Card



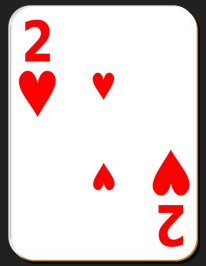






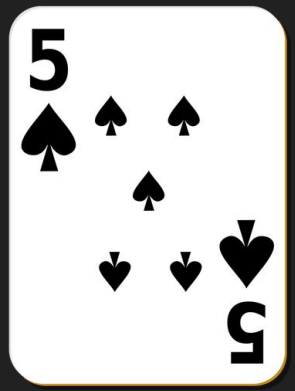
3 < 2? ✗

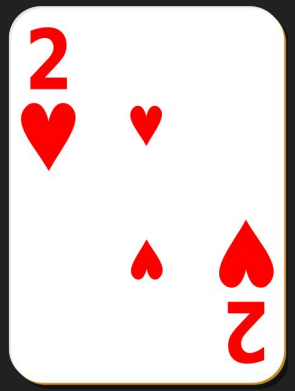
Low card:

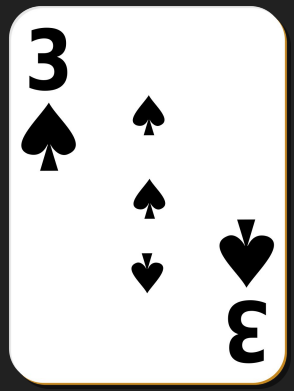


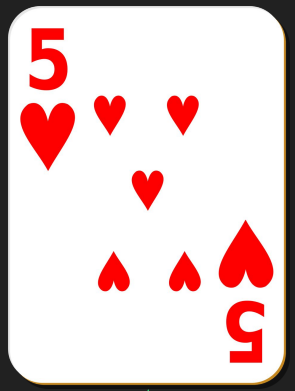
If current card < low card, make it the low card.

Recall: Finding the Lowest Card



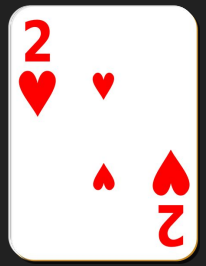






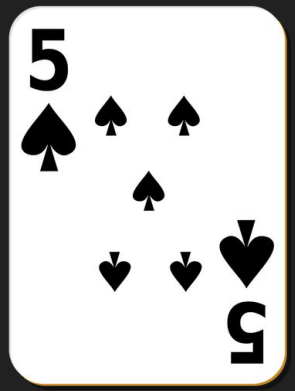
5 < 2?

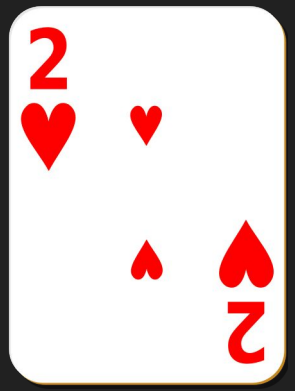
Low card:

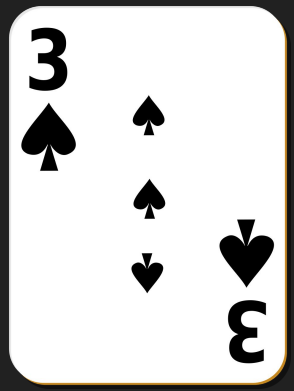


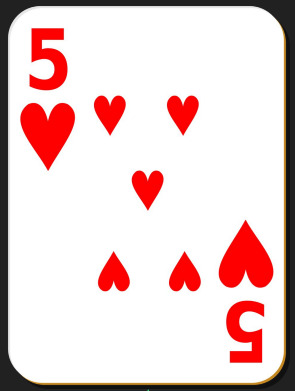
If current card < low card,
make it the low card.

Recall: Finding the Lowest Card



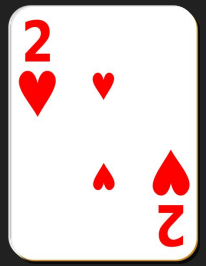






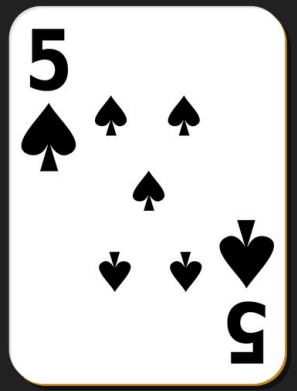
5 < 2?

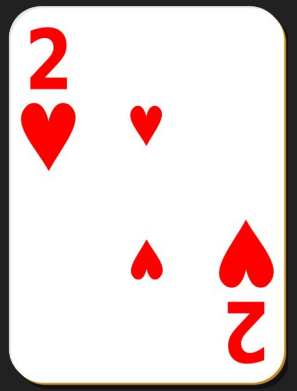
Low card:

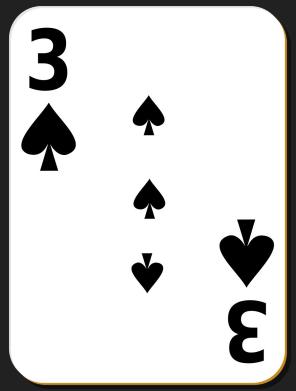


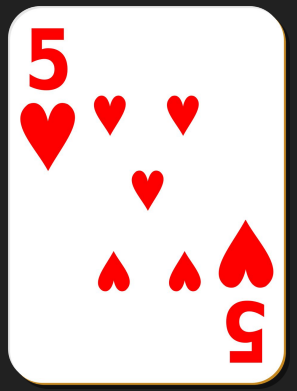
If current card < low card,
make it the low card.

Recall: Finding the Lowest Card

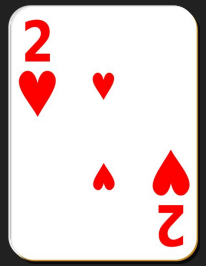








Low card:



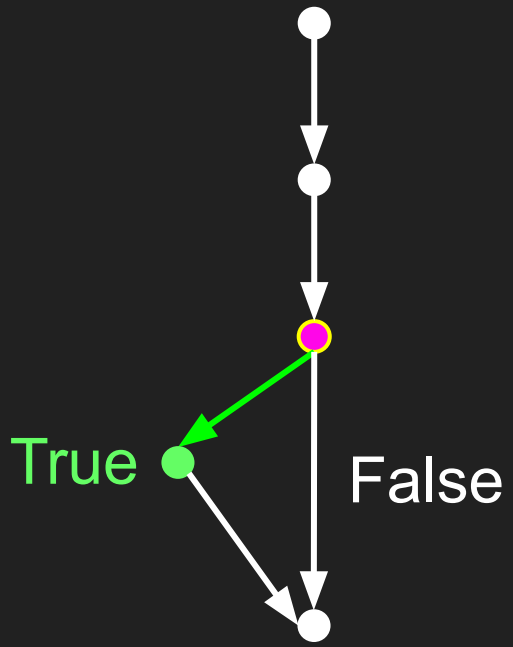
Conditional Statement

If current card < low card, make it the low card.

Conditional Statements

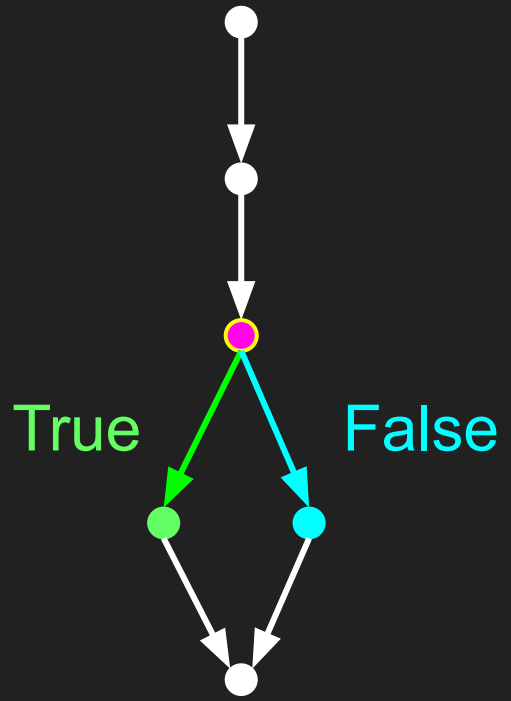
bool

```
if <something>:
    <do something>
<rest of program>
```



Conditional Statements

```
if <something>:
    <do something>
else:
    <do something else>
<rest of program>
```



Conditional Statements

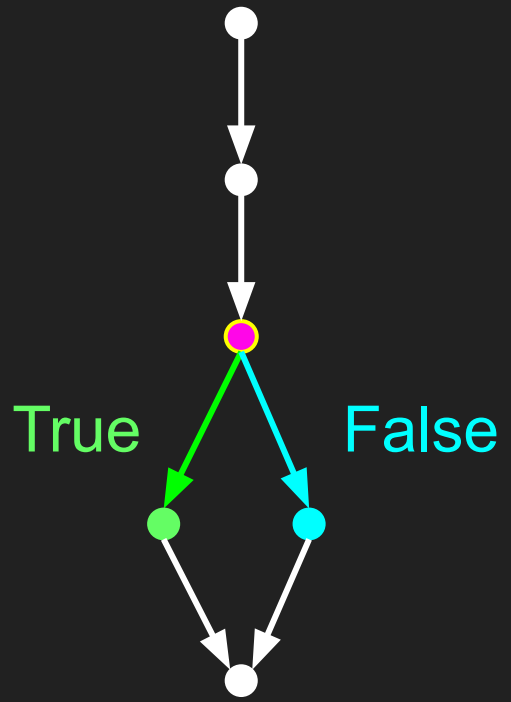
if <something>:

    <do something>

else:

    <do something else>

<rest of program>



Discussion

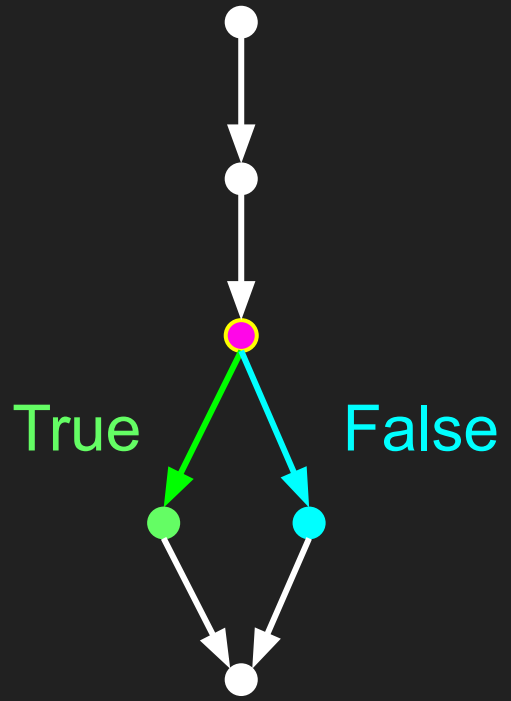
What is a decision you make in your day-to-day that you can express as an conditional (if-else) statement?

E.g. If I my assignment is due tomorrow, I start working on it. Else (it’s not due tomorrow), I procrastinate another day.
(This is bad behavior and I don’t condone it!)

Conditional Statements

if :

else:



Practice

Write a program that prints “Even” if my_number is even and “Odd” if my_number is odd.

(Hint: You will want to use % and the relational operator == from LS03)

```
1  my_number_string: str = input("Guess a number: ")
2  my_number: int = int(my_number_string)
3
4
5
6
```

elif

```
1   SECRET: int = 2
2
3   if SECRET == 10:
4       print("Correct!")
5   else:
6       if SECRET < 10:
7           print("Your guess was too low.")
8       else:
9           print("Your guess was too high.")
```

```
1   SECRET: int = 2
2
3   if SECRET == 10:
4       print("Correct!")
5   else:
6       if SECRET < 10:
7           print("Your guess was too low.")
8       else:
9           print("Your guess was too high.")
```

elif

```python
SECRET: int = 2

if SECRET == 10:
    print("Correct!")
else:
    if SECRET < 10:
        print("Your guess was too low.")
    else:
        print("Your guess was too high.")
```

```python
SECRET: int = 2

if SECRET == 10:
    print("Correct!")
elif SECRET < 10:
    print("Your guess was too low.")
else:
    print("Your guess was too high.")
```

Pause to practice:

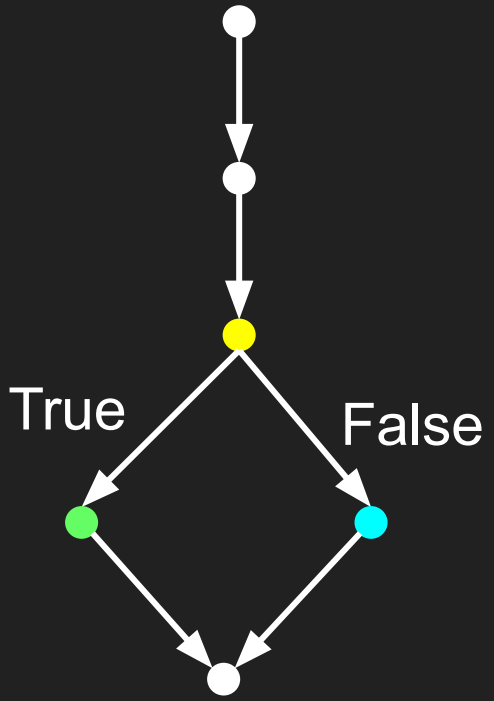
Please do the LS on Gradescope!

While Loops

First, Review

Conditionals:

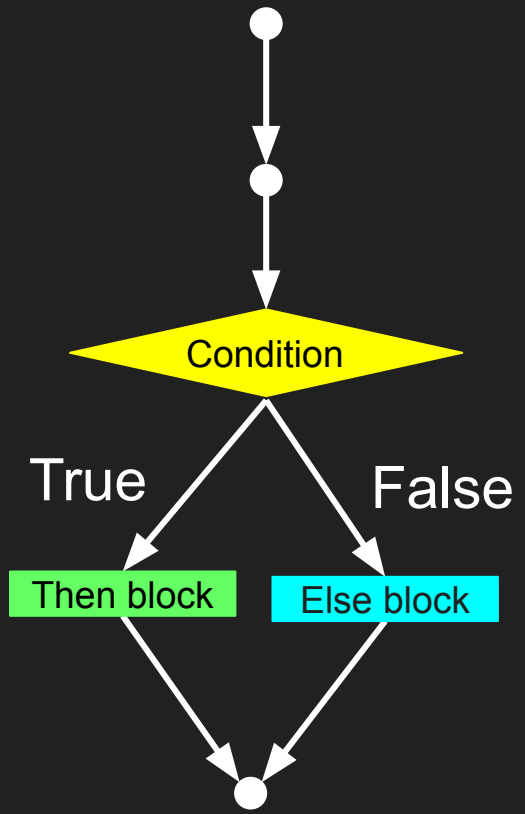
```
if <something>:
    <do something>
else:
    <do something else>
<continue program>
```



First, Review

Conditionals:

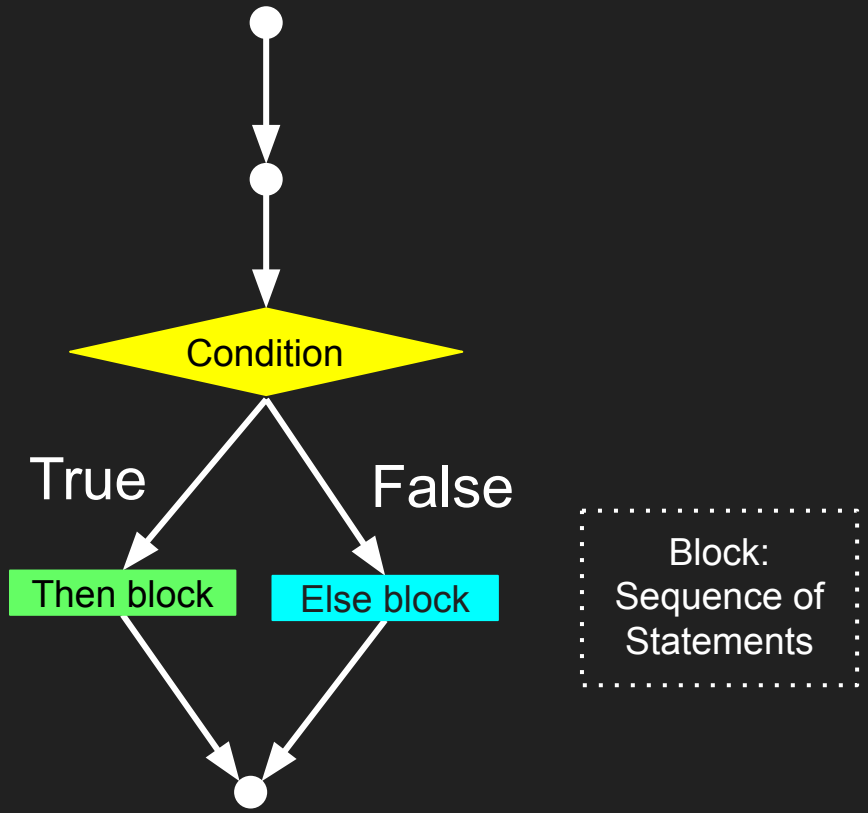
```
if <something>:
    <do something>
else:
    <do something else>
<continue program>
```



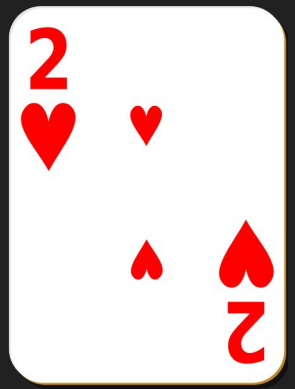
First, Review

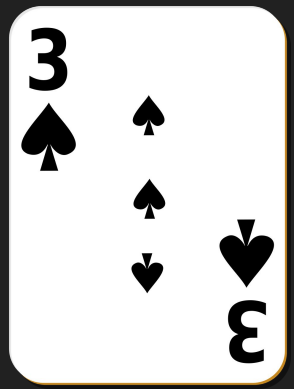
Conditionals:

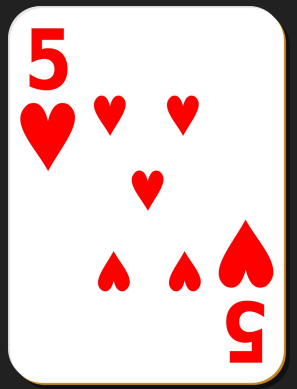
```
if <something>:
    <do something>
else:
    <do something else>
<continue program>
```



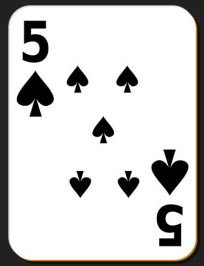
Finding the Lowest Card



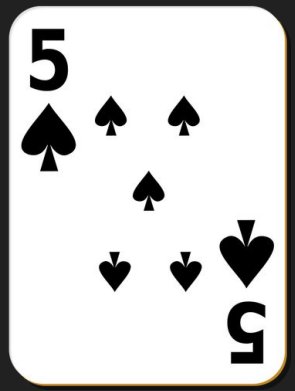


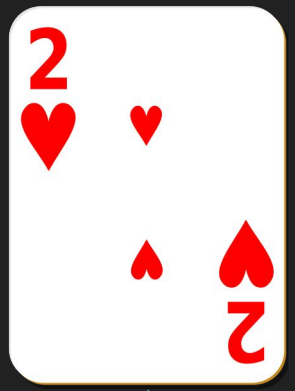


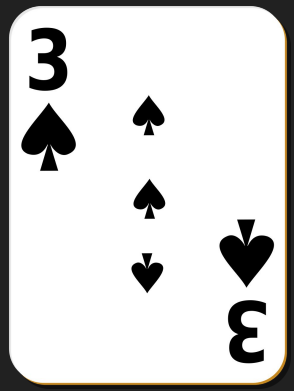
Low card:

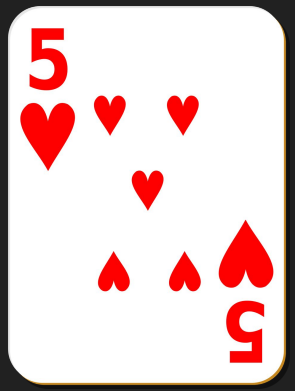


Finding the Lowest Card



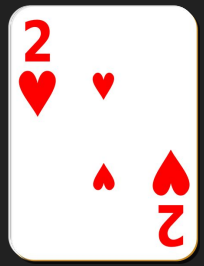




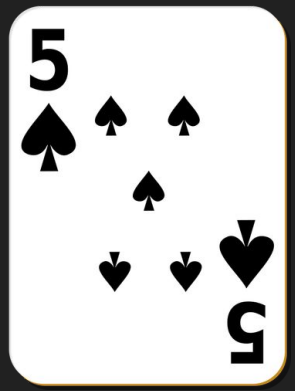


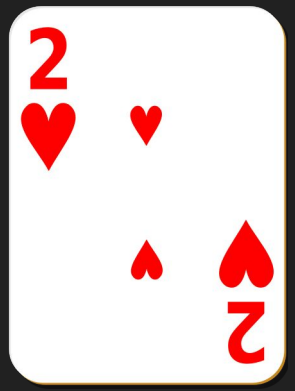
2 < 5? ✓

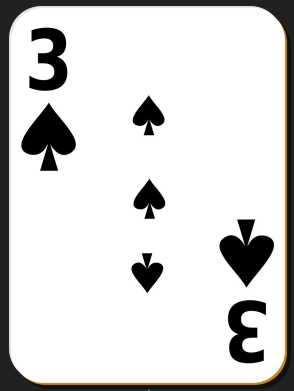
Low card:

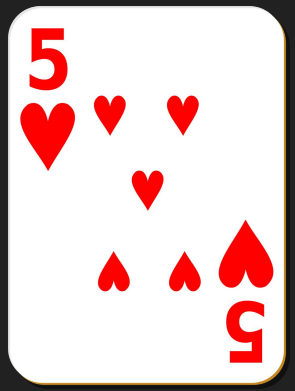


Finding the Lowest Card



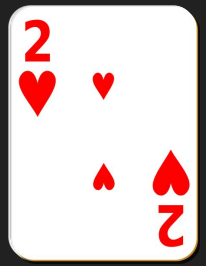




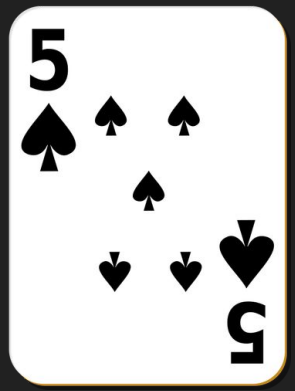


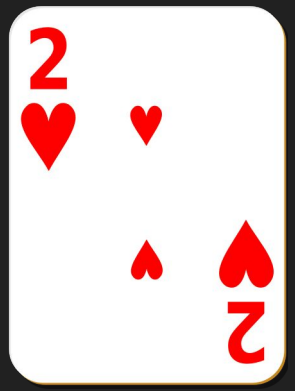
3 < 2?

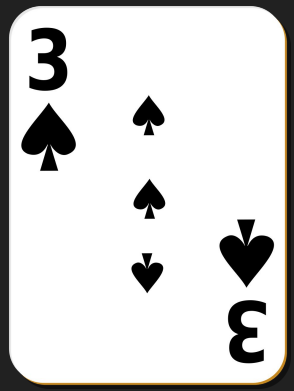
Low card:



Finding the Lowest Card

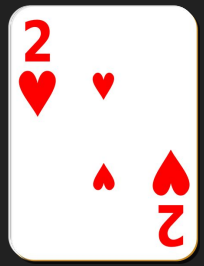






5 < 2? ✗

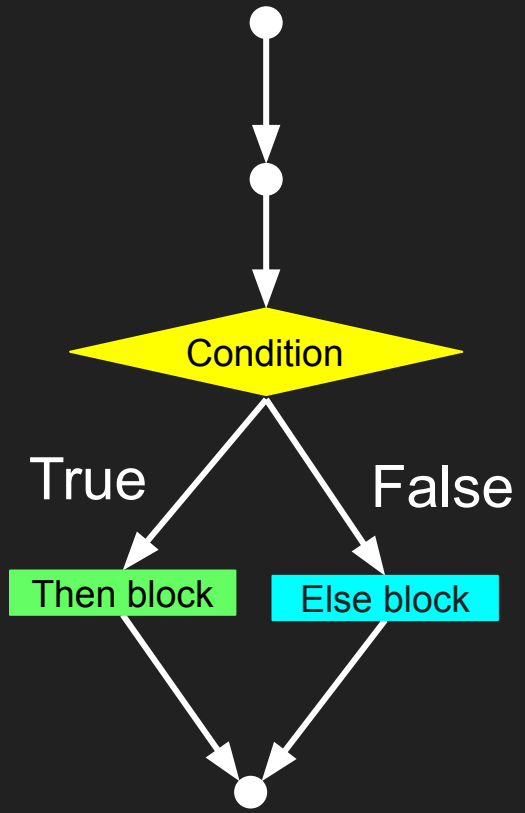
Low card:



Finding the Lowest Card

Finding the low card pseudocode:

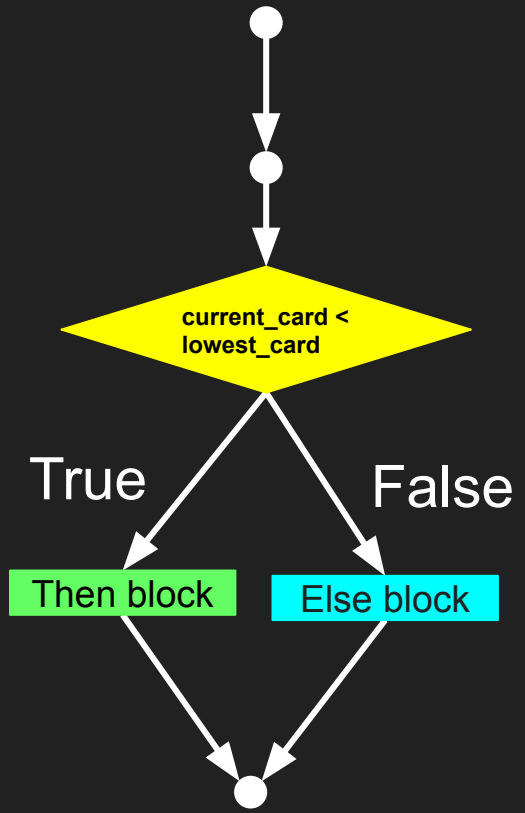
```
1 lowest_card = first card in deck
2 Repeatedly until end of deck:
3     if current_card < lowest_card:
4         lowest_card = current_card
```



Finding the Lowest Card

Finding the low card pseudocode:

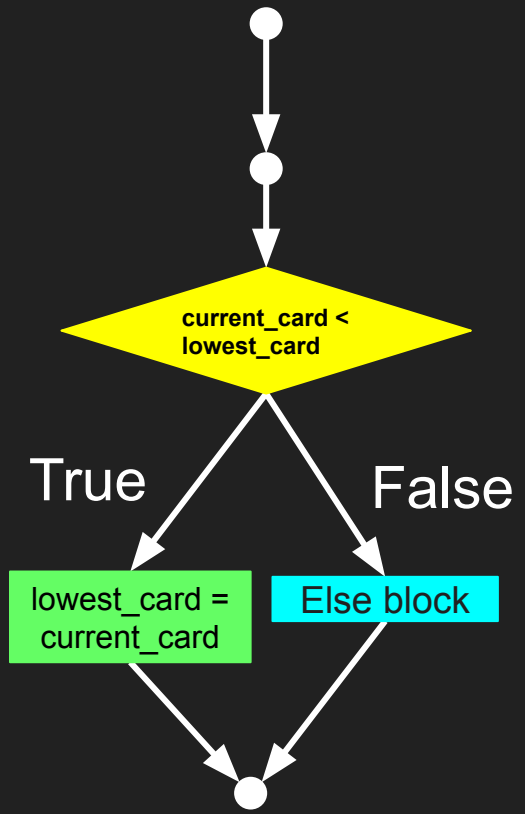
```
1 lowest_card = first card in deck
2 Repeatedly until end of deck:
3     if current_card < lowest_card:
4         lowest_card = current_card
```



Finding the Lowest Card

Finding the low card pseudocode:

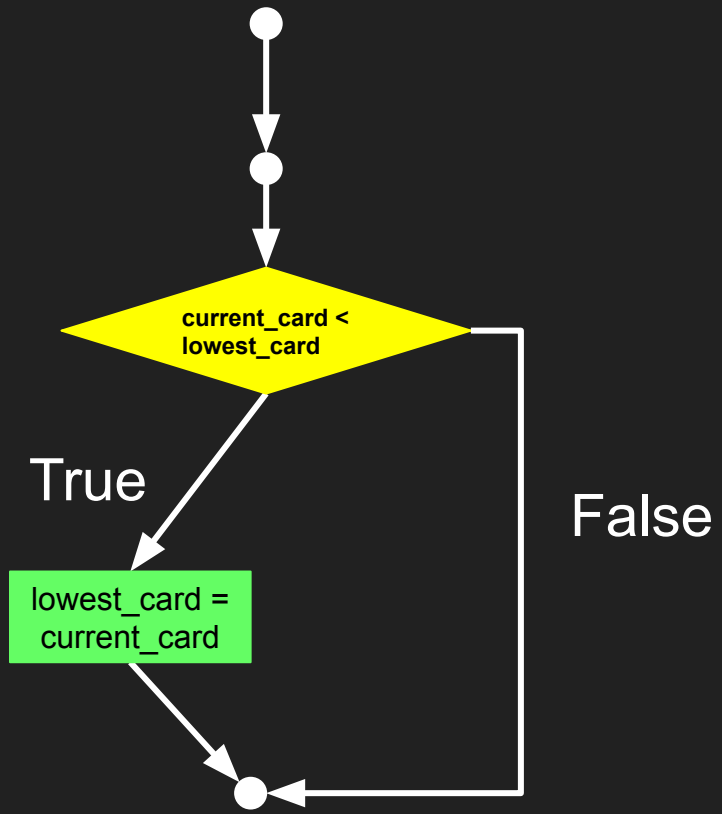
```
1 lowest_card = first card in deck
2 Repeatedly until end of deck:
3     if current_card < lowest_card:
4         lowest_card = current_card
```



Finding the Lowest Card

Finding the low card pseudocode:

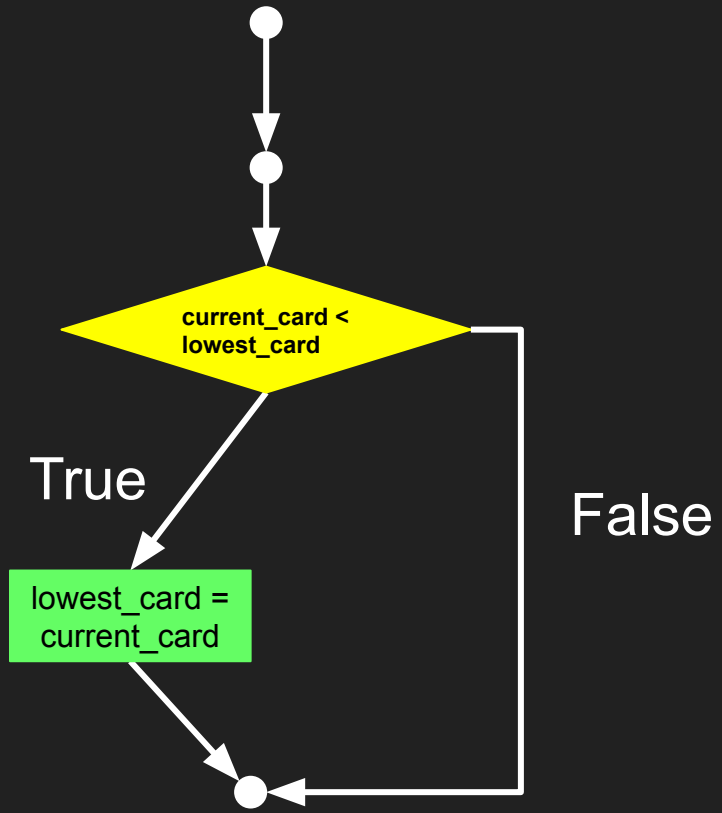
```
1 lowest_card = first card in deck
2 Repeatedly until end of deck:
3     if current_card < lowest_card:
4         lowest_card = current_card
```



Finding the Lowest Card

Finding the low card pseudocode:

```
1 lowest_card = first card in deck
2 Repeatedly until end of deck:
3       if current_card < lowest_card:
4               lowest_card = current_card
```



Loops

- Used to carry out statements in a program repeatedly an arbitrary number of times.

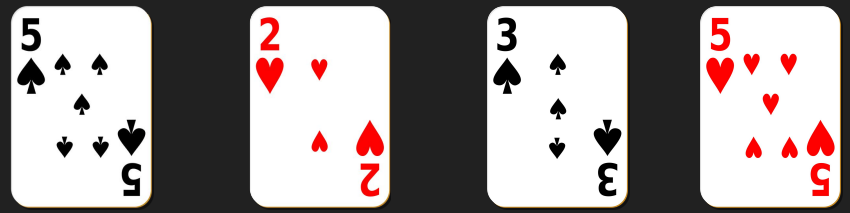
Loops

- Used to carry out statements in a program repeatedly an arbitrary number of times.

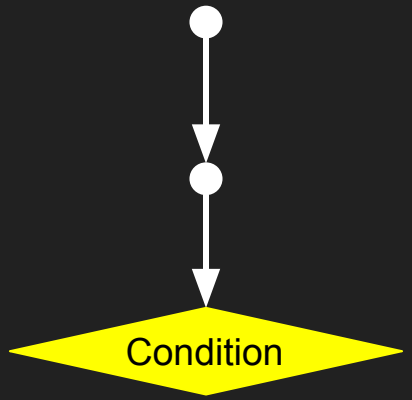
Finding the low card pseudocode:

Loop

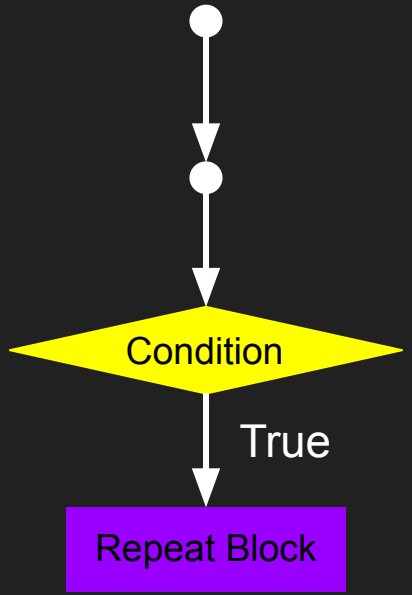
```
1 lowest_card = first card in deck
2 Repeatedly until end of deck:
3       if current_card < lowest_card:
4               lowest_card = current_card
```



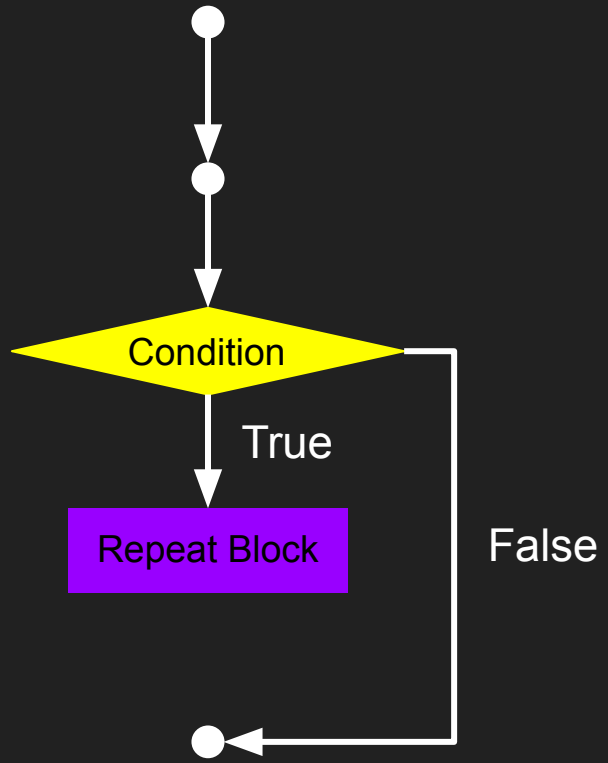
Loops



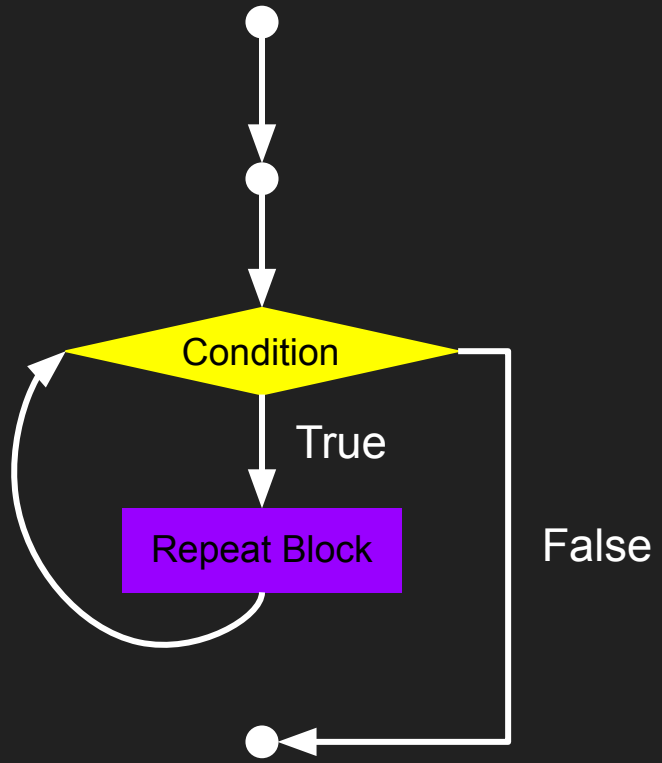
Loops



Loops

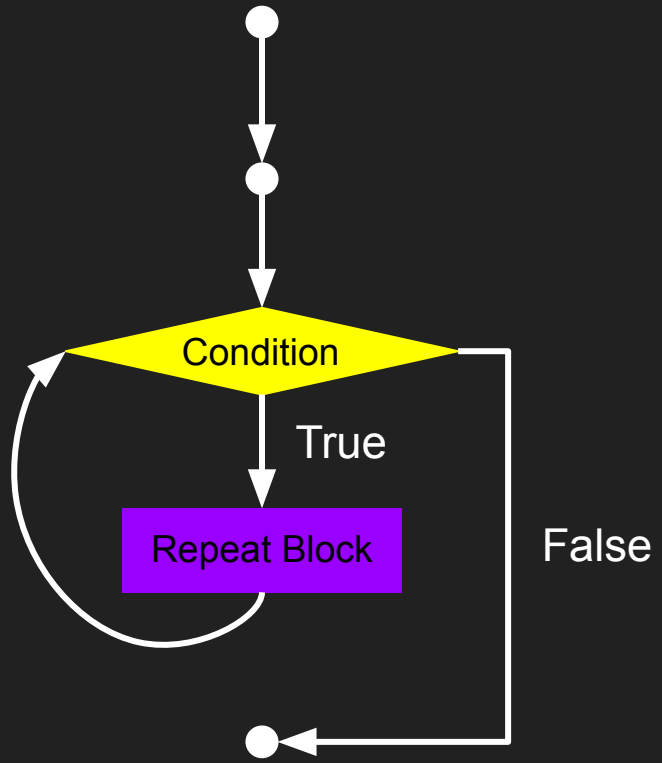


Loops



“While” Loops

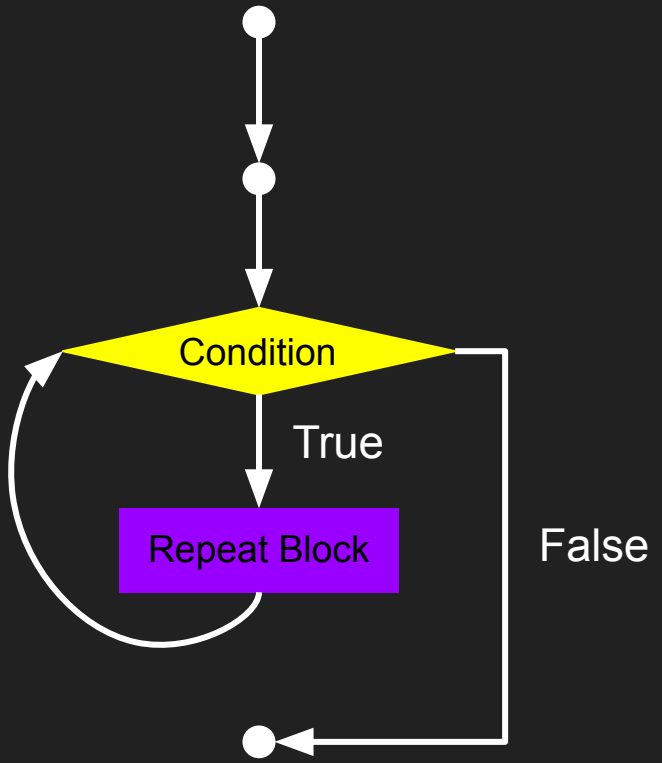
Repeat while condition is true



Loops

Finding the low card pseudocode:

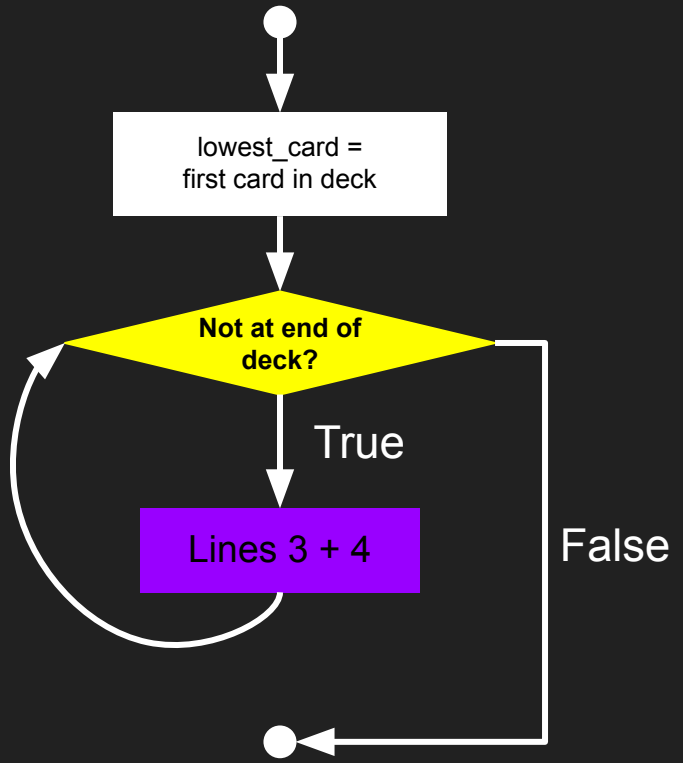
```
1 lowest_card = first card in deck
2 Repeatedly until end of deck:
3       if current_card < lowest_card:
4               lowest_card = current_card
```



Loops

Finding the low card pseudocode:

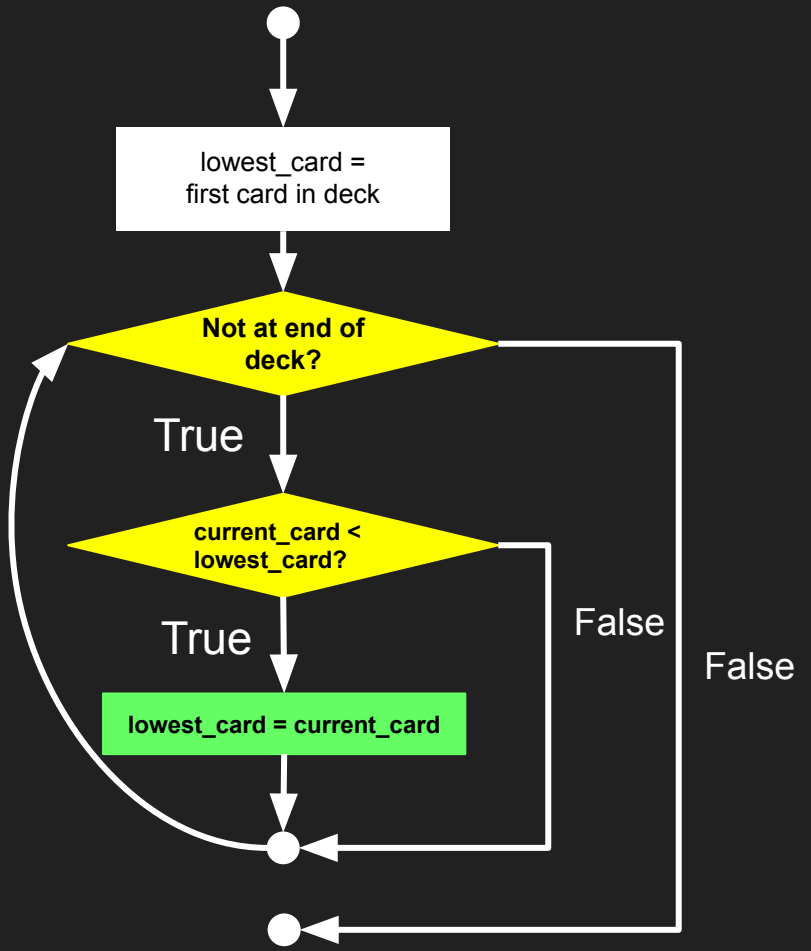
```
1 lowest_card = first card in deck
2 Repeatedly until end of deck:
3       if current_card < lowest_card:
4               lowest_card = current_card
```



Loops

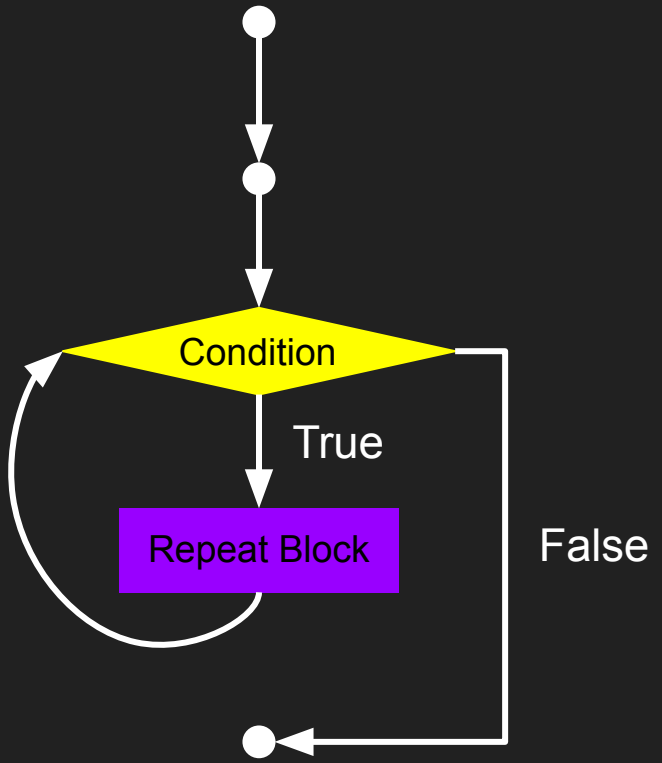
Finding the low card pseudocode:

```
1 lowest_card = first card in deck
2 Repeatedly until end of deck:
3       if current_card < lowest_card:
4               lowest_card = current_card
```



Syntax

```
while <condition>:
    <repeat action>
```



Practice (Do together)

Modify the number guessing program we just wrote to have it loop until a person guesses the correct answer.

Pause to practice:

Please do the LS on Gradescope!