

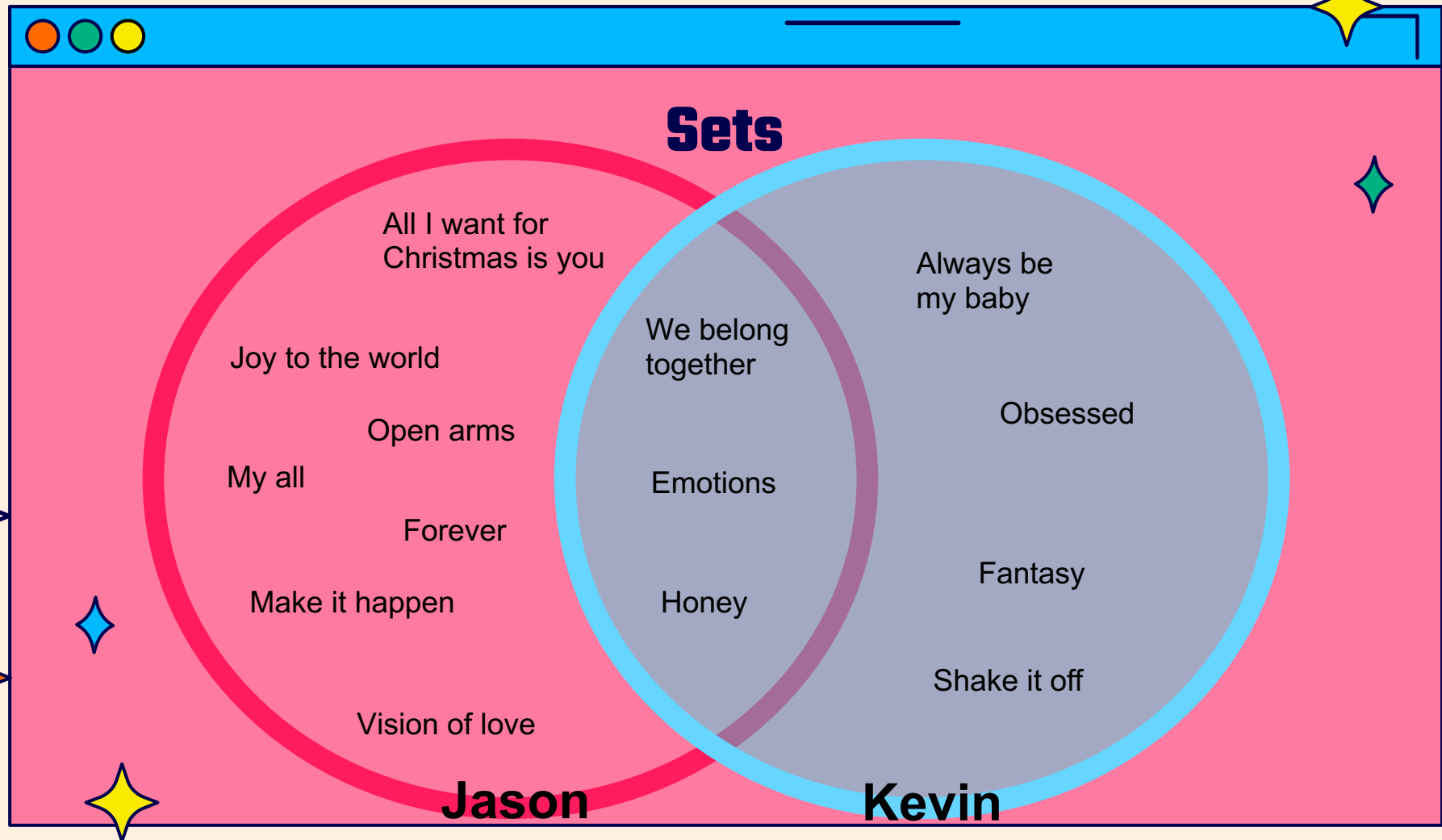


# MATLAB

Class 2: Basics part 2











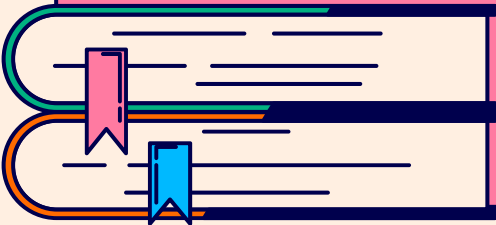
# Set operations

**Ismember** — checks if element is inside set

**Setdiff** — looks for the elements not unique to both sets

**Intersect** — returns the elements unique to both sets

**Union** — joins two sets



# Data Structures

## Tables

---

Matrix that  
can hold  
**DIFFERENT**  
types

## Structs

---

Object with  
fields used  
in object-  
oriented  
programming

## Cells

---

Container  
that can hold  
**ANY**  
information



# Tables

Keeps information in a neat fashion of different types, much like an excel sheet or a dataframe in R

- Useful when you have a matrix but need to store info of different types
- Cannot do matrix math on these
- Each column will contain info of the SAME type





# Structs

Object that contains several fields: i.e., a student has a

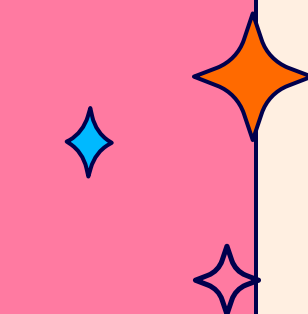
- Student.name
- Student.age
- Student.GPA
- Student.FavMariahSong
- Student.Thesis



# Cells

The most flexible of data structures in MATLAB

Holds any information you'd like in a cell

- Indexable with same rules of matrices `c{1,2}`
  - Can contain different types in each cell regardless of its neighbours (i.e., columns and rows)
- 





# Cells

Indexing a cell array is a lot like a regular array

Yet there is a key difference between `c(1,2)` and `c{1,2}`

The latter indexes the **contents** of the cell, the former indexes the **cell** itself

Note: you can index an array after indexing a cell `c{1,1}(1)`







# Cell Operators

To help visualize your cell structure and the contents it holds use **cellpolt()**





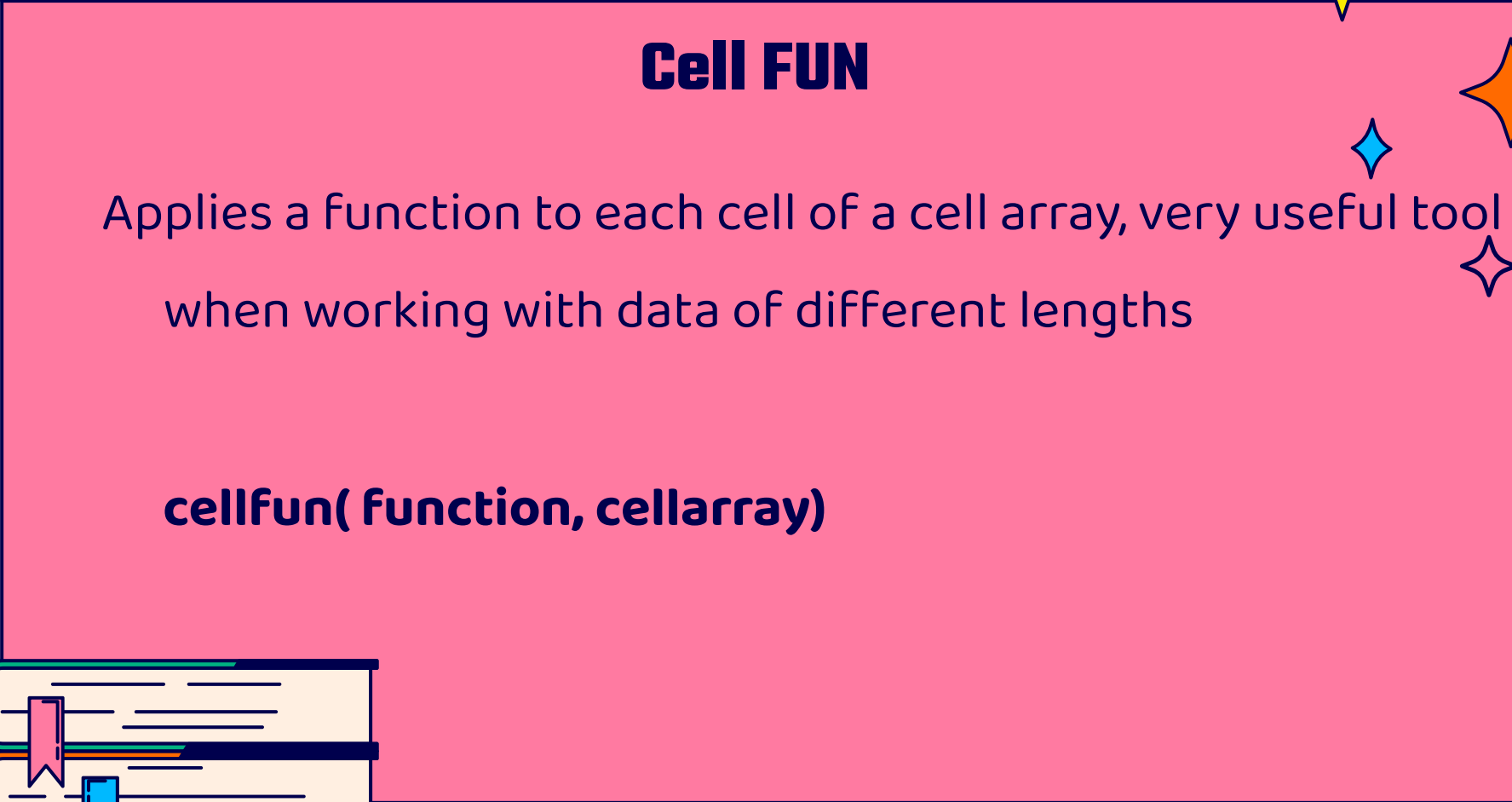
You can also convert between cells, structs, matrices, etc given that data conversion is possible





# Cell FUN

Applies a function to each cell of a cell array, very useful tool  
when working with data of different lengths



**cellfun( function, cellarray)**





## Conditionals and flow of logic

Sometimes we want something to happen only **IF** a criterion is true or a specific **CASE** is met

For example:

we only want to include subjects **IF** their Ids are odd

we only want to warn users in **CASE** of an error



## REMINDER: Boolean Operators

How can we ask the computer a question:

Is equal to ==

Is greater than >

Is less than <

Is NOT equal to ~=

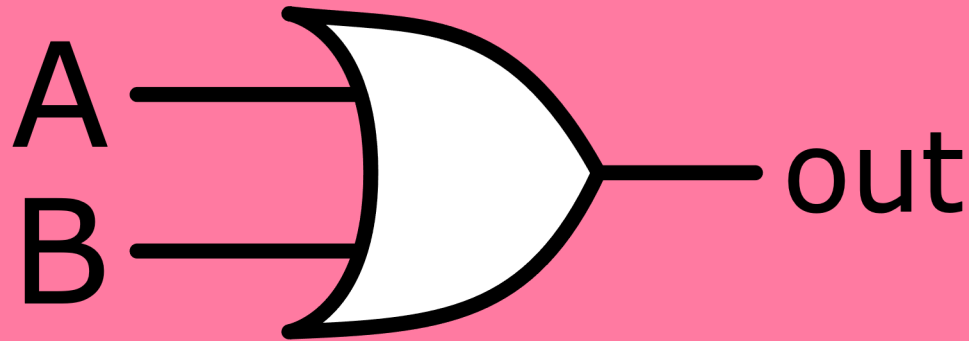
The OR operator ||

The AND operator &&



# Boolean Operators

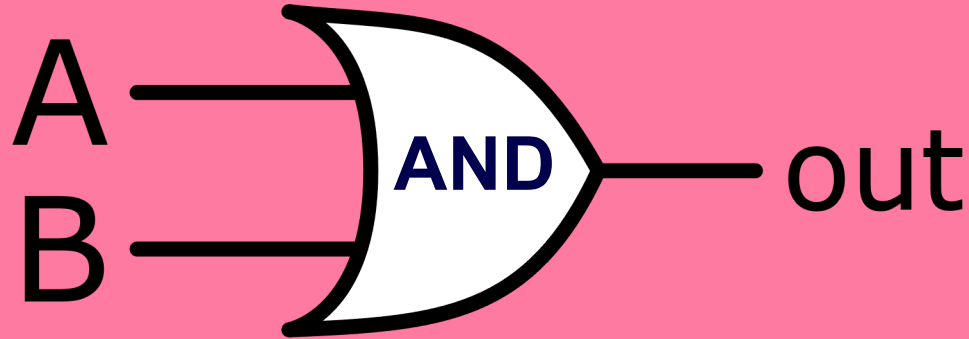
Two very special operators: **AND** and **OR**





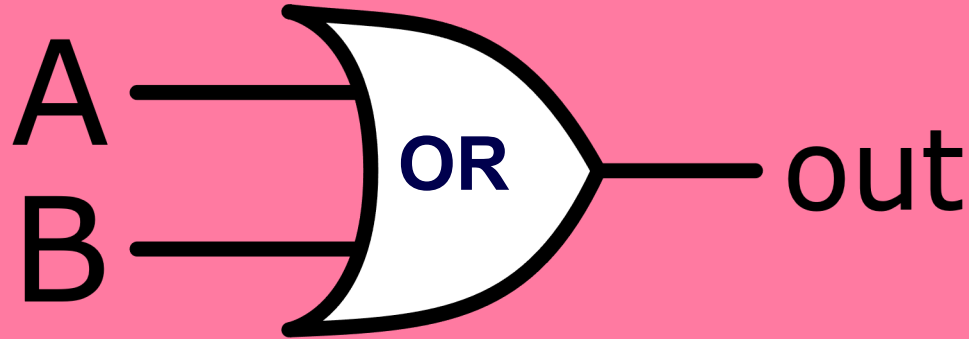
# Boolean Operators

Two very special operators: **AND** and **OR**



# Boolean Operators

Two very special operators: **AND** and **OR**



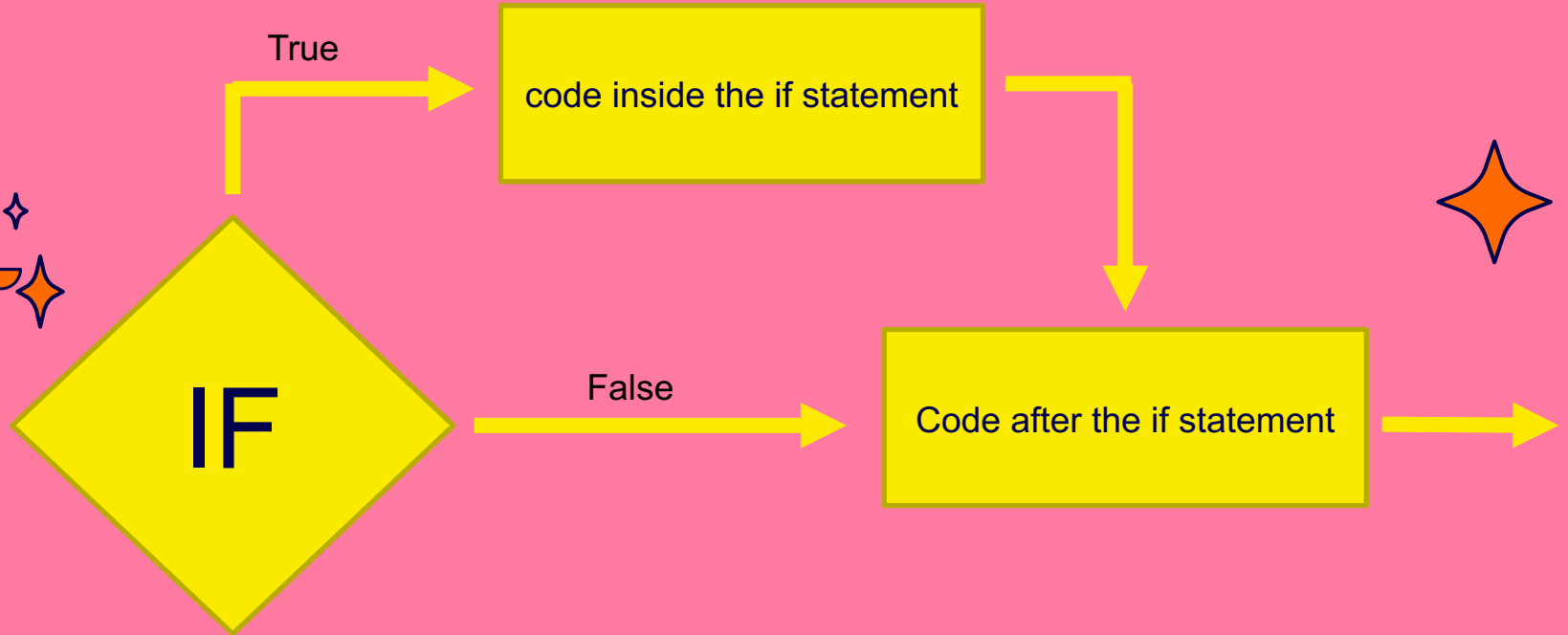


# IF & SWITCH statement

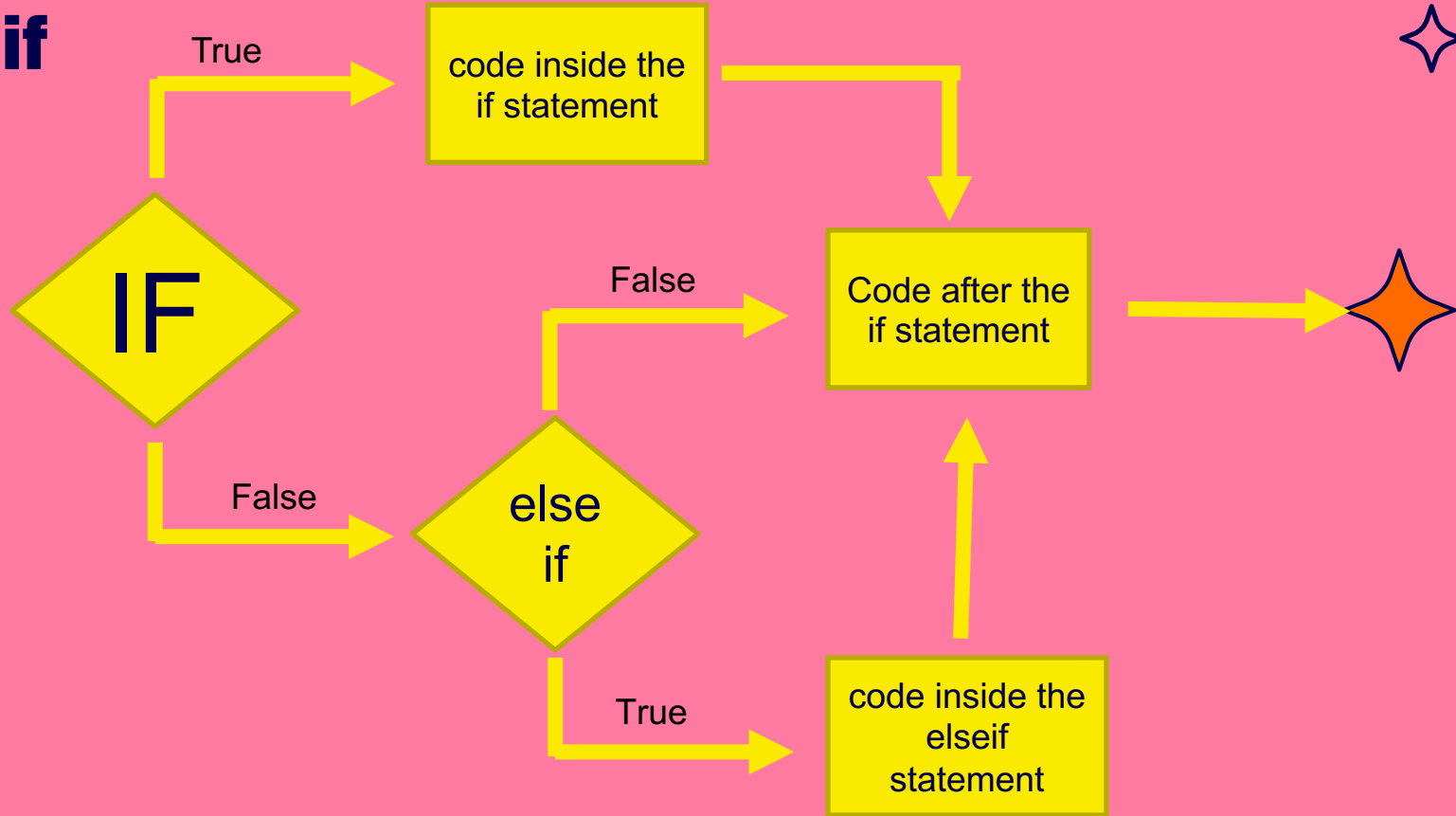
These are the gate keepers/ decision makers of your  
code

These functions allow you to branch your code  
depending on conditions

# IF & SWITCH statement



# Elseif





# SWITCH statement

Useful when there are a finite number of acceptable  
inputs that you want to check the value of

Works exactly like an if but with cases, if a **case** is not  
met you move on to the next



# LOOPS

We also might want to repeat lines of code several times

Instead of copying and pasting code 100 times we can use loops



# For LOOPS

For variable\_name = values to iterate over

Do STUFF in here.....

end





# While LOOPS

```
while      conditional statement
```

```
Do STUFF in here.....
```

```
end
```

# While LOOPS

**while**                      conditional statement

Do STUFF in here.....

**end**



# While LOOPS

```
while conditional statement
```

```
Do STUFF in here.....
```

```
end
```

# While LOOPS

**while**                      conditional statement

Do STUFF in here.....

**end**

# While LOOPS

```
while      conditional statement
```

Remember you need to update the value  
of the conditional such that it will  
terminate after a given point

```
end
```

# While LOOPS

```
while      conditional statement
```

If you do not update the conditional you  
will have an **infinite loop**

```
end
```

The diagram illustrates a while loop structure within a pink rectangular frame. At the top, a blue header bar contains three colored circles (orange, green, yellow) on the left and a yellow star on the right. The main content area is pink. A blue box at the top contains the text 'while' followed by 'conditional statement'. Below this is a large yellow box containing the text 'If you do not update the conditional you will have an infinite loop'. At the bottom is another blue box containing the text 'end'. A dark blue arrow on the left points downwards from the 'while' box to the 'end' box. A dark blue arrow on the right points upwards from the 'end' box back to the 'while' box, indicating a loop. Several decorative stars (yellow, green, orange, and blue) are scattered around the diagram.



## Try catch

Statements that control the flow of the code

Very useful when debugging or trying to 'foolproof'  
your code

Will try an assignment or function call and return an  
error. Does not stop the execution of your code!