Loops

Infinite Loops

- The simplest way of looping is using the `loop{}` key word
    - This loops infintely unless there is a break statement
    - Break statements and continue work the same as previous languages
    - Example:

    ```
    let mut x = 1;
    loop{
        if x == 11{
            break;
        }
        println!("{x:?}");
        x += 1;
    }
    ```

- While loops, similar to the ones in normal languages
    - Takes a condition and runs until condition is true
    - Example:

    ```
    let mut n = 1;
    while (n < 50){
        if (n%5 == 0){
            println!("{n:?}");
        }
        else{
            continue;
        }

        n+=1;
    }
    ```

- For loops similar to the ones in normal languages
    - It an iterative loop that breaks when it is out of range
    - Iterating over intergers
        - Range is done by the `..` operator so 1->10 is `1..11`
        - Example:

        ```
        for i in 1..11{
            println!("{i:?}");
        ```

```
}
```

- Define any type of range
  - `let numbers = 32..51;`
  - `let numbers = [2,3,5,7,8,11];`
  - `let lst = ["cat", "dog", "cow"];`
  - Example:

    ```
    let lst = ["cat", "dog", "cow"];
    for i in lst.iter(){
        println!("{i:?}");
    }
    ```

  - You should use the .iter(), this is an ownership issue but for now keep it in
  - If you want to know the index of the item of a list or range you need to use `.enumerate()`
    - This returns a tuple of `(index, value)`
    - Example:

      ```
      let lst = ["wolf", "tiger", "fox", "elephant"];
      for (i, val) in lst.iter().enumerate(){
          if i % 2 == 0 {
              println!("{:?} -> {:?}",i,val);
          }

      }
      ```