INTERMEDIATE R

while loop

while loop

```
while(condition) {
  expr
}
```

```
> ctr <- 1

> while(ctr <= 7) {
```

while loop

```
while(condition) {
  expr
}
```

```
> ctr <- 1

> while(ctr <= 7) {
    print(paste("ctr is set to", ctr))
```

while loop

```
while(condition) {
  expr
}
```

```
> ctr <- 1

> while(ctr <= 7) {
    print(paste("ctr is set to", ctr))
    ctr <- ctr + 1 add another line of code to inform R that we want to increment the ctr variable on every run
  }
```

while loop

```
while(condition) {
  expr
}
```

```
> ctr <- 1
> while(ctr <= 7) {
    print(paste("ctr is set to", ctr))
    ctr <- ctr + 1
  }
[1] "ctr is set to 1"
```

ctr: 1

TRUE

increment ctr

while loop

```
while(condition) {
  expr
}
```

```
> ctr <- 1
> while(ctr <= 7) {
    print(paste("ctr is set to", ctr))
    ctr <- ctr + 1
  }
[1] "ctr is set to 2"
```

TRUE

increment ctr

ctr: 2

while loop

```
while(condition) {
  expr
}
```

```
> ctr <- 1
> while(ctr <= 7) {
    print(paste("ctr is set to", ctr))
    ctr <- ctr + 1
  }
[1] "ctr is set to 7"
```

TRUE

increment ctr

ctr: 7

while loop

```
while(condition) {
  expr
}
```

```
> ctr <- 1
```

FALSE

```
> while(ctr <= 7) {
    print(paste("ctr is set to", ctr))
    ctr <- ctr + 1
  }
```

ctr: 8

No printout!

while loop

```
> ctr <- 1

> while(ctr <= 7) {
    print(paste("ctr is set to", ctr))
    ctr <- ctr + 1
  }
[1] "ctr is set to 1"
[1] "ctr is set to 2"
...
[1] "ctr is set to 7"

> ctr
[1] 8
```

infinite while loop

```
> ctr <- 1

> while(ctr <= 7) {
    print(paste("ctr is set to", ctr))
    ctr <- ctr + 1 suppose we remove this line
  }
[1] "ctr is set to 1"
[1] "ctr is set to 1"
[1] "ctr is set to 1"
[1] "ctr is set to 1"
[1] "ctr is set to 1"
[1] "ctr is set to 1"
[1] "ctr is set to 1"
[1] "ctr is set to 1"
...
```

break statement

```
> ctr <- 1
> while(ctr <= 7) {
    if(ctr %% 5 == 0) {
      break
    }
    print(paste("ctr is set to", ctr))
    ctr <- ctr + 1
  }
[1] "ctr is set to 1"
[1] "ctr is set to 2"
[1] "ctr is set to 3"
[1] "ctr is set to 4"
```

TRUE

Break if ctr is a 5-fold

while loop stops if ctr is 5: no more printouts

The break statement simply breaks out of the while loop: when R finds it, it abandons the currently active while loop.

INTERMEDIATE R

Let’s practice!

INTERMEDIATE R

for loop

for loop

```
for(var in seq) {
  expr
}
```

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> cities
[1] "New York"    "Paris"   ...   "Cape Town"
```

for loop

```
for(var in seq) {
  expr
}
```

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(var in seq) {
    expr
  }
```

for loop

```
for(var in seq) {
  expr
}
```

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    expr
  }
```

for loop

```
for(var in seq) {
  expr
}
```

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    print(city)
  }
```

for loop

```
for(var in seq) {
  expr
}
```

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    print(city)
  }
[1] "New York"
```

city: "New York"

for loop

```
for(var in seq) {
  expr
}
```

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    print(city)
  }
[1] "Paris"
```

city: "Paris"

for loop

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    print(city)
  }
[1] "New York"
[1] "Paris"
[1] "London"
[1] "Tokyo"
[1] "Rio de Janeiro"
[1] "Cape Town"
```

for loop over list

```
> cities <- list("New York", "Paris",
                 "London", "Tokyo",
                 "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    print(city)
  }
[1] "New York"
[1] "Paris"
[1] "London"
[1] "Tokyo"
[1] "Rio de Janeiro"
[1] "Cape Town"
```

break statement

```
> cities <- list("New York", "Paris",
                 "London", "Tokyo",
                 "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    if(nchar(city) == 6) {
      break
    }
    print(city)
  }
```

break statement

```
> cities <- list("New York", "Paris",
                 "London", "Tokyo",
                 "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    if(nchar(city) == 6) {
      break
    }
    print(city)
  }
[1] "New York"
```

city: "New York"

FALSE

break statement

```
> cities <- list("New York", "Paris",
                 "London", "Tokyo",
                 "Rio de Janeiro", "Cape Town")

> for(city in cities) {                          city: "Paris"
    if(nchar(city) == 6) {   FALSE
      break
    }
    print(city)
  }
[1] "Paris"
```

break statement

```
> cities <- list("New York", "Paris",
                 "London", "Tokyo",
                 "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    if(nchar(city) == 6) {
      break
    }
    print(city)
  }
```

city: "London"

TRUE

for loop abandoned, no printout

break statement

```
> cities <- list("New York", "Paris",
                 "London", "Tokyo",
                 "Rio de Janeiro", "Cape Town"))

> for(city in cities) {
    if(nchar(city) == 6) {
      break
    }
    print(city)
  }
[1] "New York"
[1] "Paris"
```

next statement

```
> cities <- list("New York", "Paris",
                 "London", "Tokyo",
                 "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    if(nchar(city) == 6) {
      next
    }
    print(city)
  }
[1] "New York"
[1] "Paris"
[1] "Tokyo"
[1] "Rio de Janeiro"
[1] "Cape Town"
```

next: *skip* to next iteration

"London" is not printed!

The next statement skips the remainder of the code inside the for loop and proceeds to the next iteration.

for loop: v2

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(city in cities) {
    print(city)
  }
```

for loop: v2

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(i in 1:length(cities)) {
    print(city) change here
  }
```

1:6 == c(1, 2, 3, 4, 5, 6)

Instead of iterating over the cities, we can manually create a looping index ourselves.

for loop: v2

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(i in 1:length(cities)) {
    print(cities[i])
  }
[1] "New York"
[1] "Paris"
[1] "London"
[1] "Tokyo"
[1] "Rio de Janeiro"
[1] "Cape Town"
```

for loop: v2

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")

> for(i in 1:length(cities)) {
    print(paste(cities[i], "is on position",
                i, "in the cities vector."))
  }
[1] "New York is on position 1 in the cities vector."
[1] "Paris is on position 2 in the cities vector."
[1] "London is on position 3 in the cities vector."
[1] "Tokyo is on position 4 in the cities vector."
[1] "Rio de Janeiro is on position 5 in the cities vector."
[1] "Cape Town is on position 6 in the cities vector."
```

for loop: wrap-up

```
> cities <- c("New York", "Paris",
              "London", "Tokyo",
              "Rio de Janeiro", "Cape Town")
```

```
> for(city in cities) {
    print(city)
  }
```

+ Concise
+ Easy to read
- No access to looping index

```
> for(i in 1:length(cities)) {
    print(cities[i])
  }
```

- Harder to read and write
+ More versatile

INTERMEDIATE R

Let’s practice!