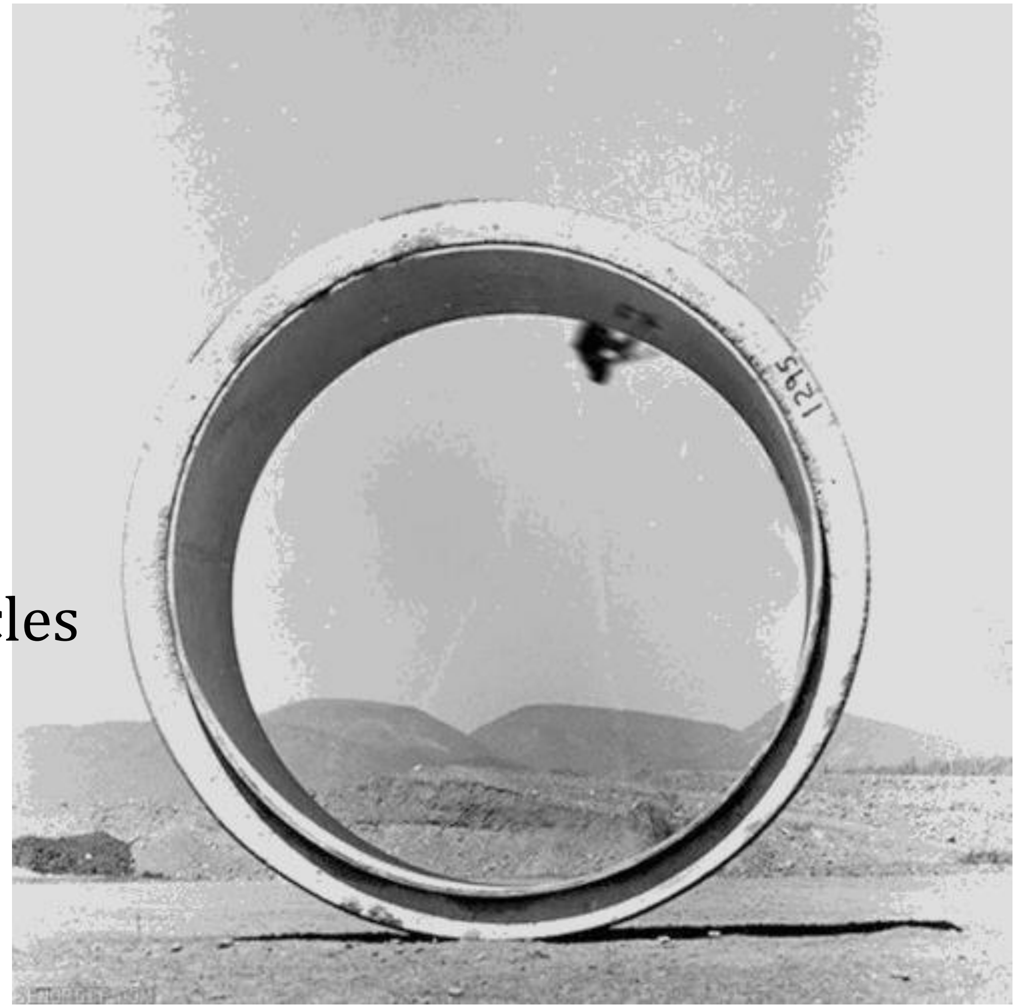
# Loops

Going around in Circles



## While Loop

while(<condition>) <statement;>

- <condition> : Any expression evaluated as true or false
- <statement> : Any valid statement or block of statements

- Statement (or block) is re-executed as long as the condition is true
- If condition is false to start with, statement is never executed!

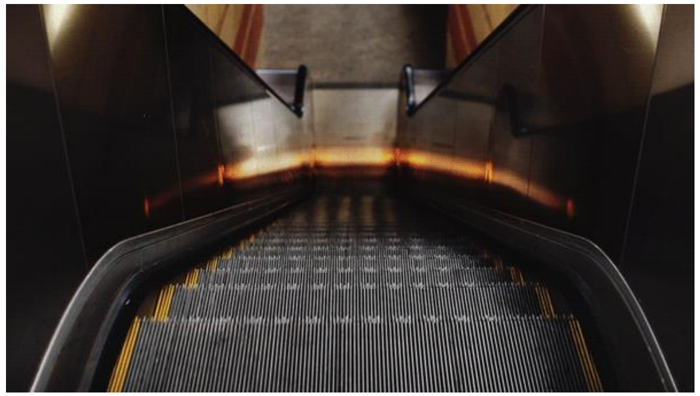
## Example While Loop

```
int temp=check_temp();
while(temp<100) {
        add_heat();
        temp=check_temp();
}
/* temperature reached 100... water should boil */
```

## Example While Loop

```
int count=100;
while(count>0) {
        int result=experiment(count);
        printf("%3i : ",count);
        int j=0;
        while(j<result) {  printf("*"); j++; }
        printf("\n");
        count--;
}
```

```
100:
 99: *
 98: *
 97: **
 96: *
 95: **
 94: ***
 93: *****
 92: ******
 91: **********
 90: ***********
 89: ********
 88: ******
 87: ****
 86: ***
 85: **
 84: *
…
```



# Example While Loop

```
int count=100;
while(count>0) {
        int result=experiment(count);
        printf("%3i : ",count);
        int j=0;
        while(j<result) {  printf("*"); j++; }
        printf("\n");
        count--;
}
```

Initialization

Condition

Increment

```
100:
 99: *
 98: *
 97: **
 96: *
 95: **
 94: ***
 93: *****
 92: ******
 91: **********
 90: ***********
 89: ********
 88: ******
 87: ****
 86: ***
 85: **
 84: *
...
```

## for loop

```
for (<init>;<condition>;<iteration>) <statement;>
```

- <init> : Statement executed before loop starts
- <condition> : Check to see if loop should continue
- <iteration> : Statement executed after loop, before condition
- <statement> : Statement or block that makes up the body of the loop

## Example for Loop

```
int count;
for(count=100;count>0;count--) {
        int result=experiment(count);
        printf("%3i : ",count);
        int j;
        for(j=0;j<result;j++) printf("*");
        printf("\n");
}
```

```
100:
 99: *
 98: *
 97: **
 96: *
 95: **
 94: ***
 93: *****
 92: ******
 91: **********
 90: ***********
 89: ********
 88: ******
 87: ****
 86: ***
 85: **
 84: *
…
```

## More Example For Loops

```
for(data=get_first(); more_data(); data=get_next()) {
        process(data);
}

for(i=0,sum=0; i<100;) sum+=experiment(i++);
average=sum/100;
```

## Infinite Loops

```
int i=get_max();
while(i!=0) {
        process(i);
        i--;
}

for(i=0; i<10; i++) {
        process(i); i=3;
}

x=1;
while(x) { do_stuff(x); }
```

## Breaking out of loops early

- break; statement leaves innermost loop (while/for/switch)
  - No checking of <condition>
  - No increment in for loop

```
for(i=0;i<100;i++) {
        result=experiment(i);
        if (result<0) break; /* Something bad happened */
}
```

## Early Iteration of Loops

- continue; statement ends this iteration of the loop
  - In for loops, iteration statement executed again
  - condition re-evaluated
  - If condition is true, next iteration of the loop starts

```
for(count=0; count<100; count++) {
        if (0==count%7) continue; // skip every 7th experiment
        result=experiment(count);
        …
}
```

# Resources

- Programming in C, Chapter 5
- Wikipedia: Conditional (computer programming) (https://en.wikipedia.org/wiki/Conditional_(computer_programming))