Variables

```
type variable_name(s)
----------------------
char grade;
float x, y, z;
int score, num_of_teams;
```

Variables

```
type variable_name(s)
----------------------
char grade = 'A';
float x, y, z;
int score = 7, teams = 4;
```

Variables

```
type variable_name(s)
----------------------
char grade = 'A';
float x, y, z;
// current score of game
int score = 7;
// number of teams on the field
int teams = 4;
```

Variables

- meaningful variable names
- in loops, a single letter is ok
- consistent initialization

```
int quarters, dimes = 4;
```

Loops

Scratch

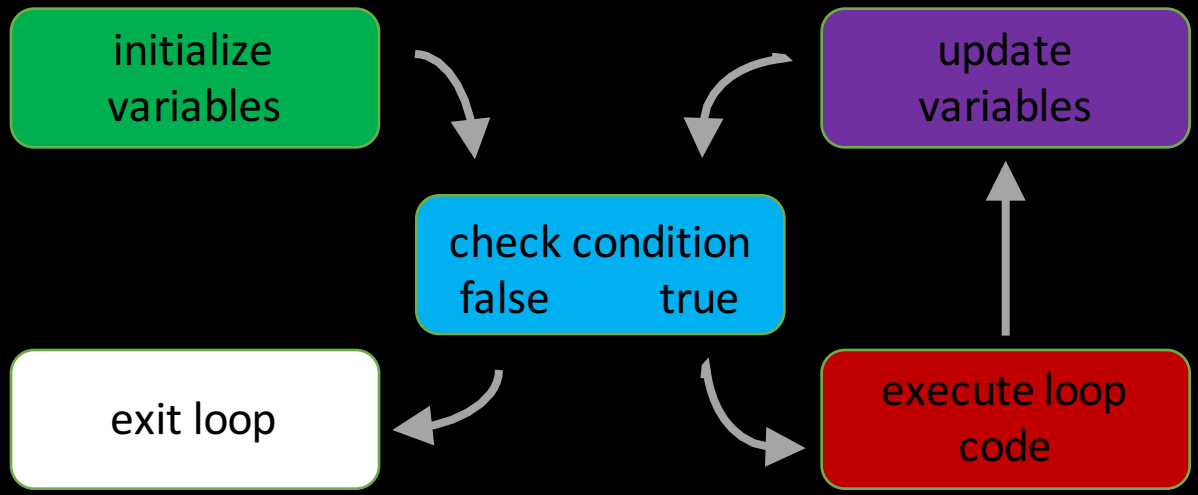
repeat (5)
repeat until key “space” pressed
forever

C

for
while
do while

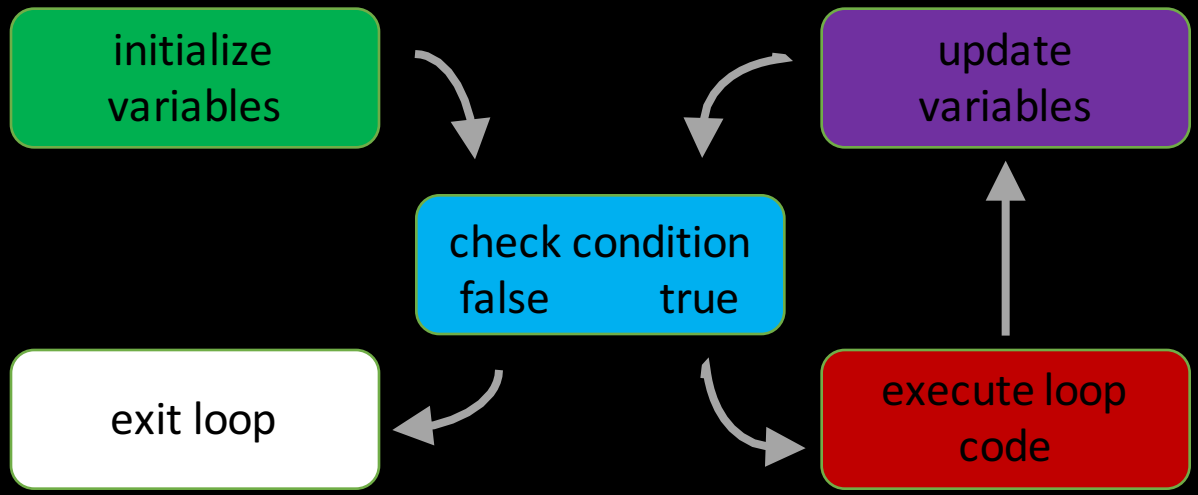
For loops

```
for (initialization; condition; update)
{
    execute this code;
}
```



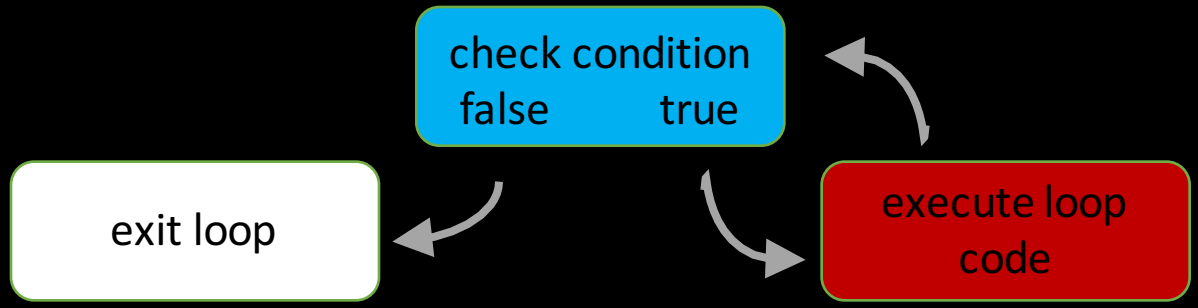
For loops

```
for (int i = 0; i < 10; i++)
{
    printf("This is CS50!\n");
}
```



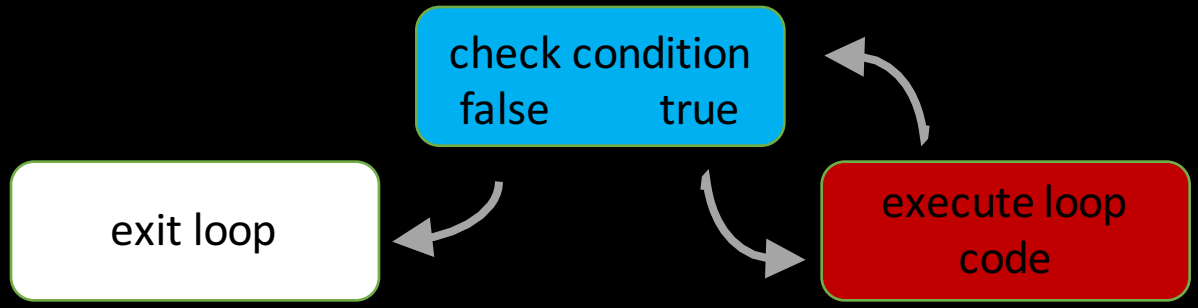
While loops

```
while (condition)
{
    execute this code;
}
```



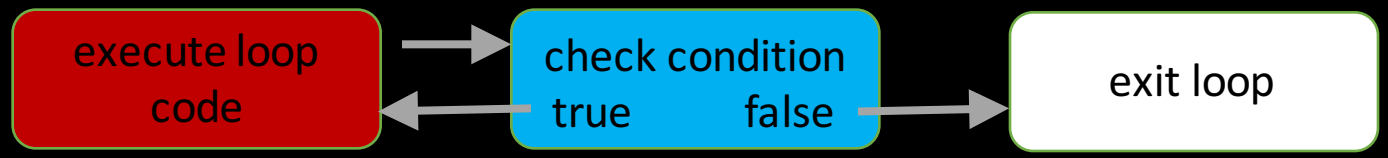
While loops

```
while (true)
{
    printf("And in that moment…\n");
}
```



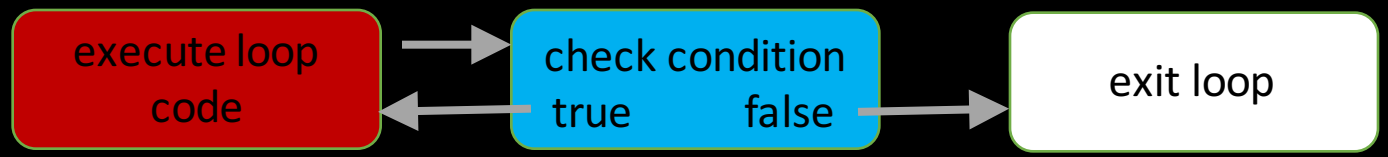
Do while loops

```
do
{
    execute this code;
}
while (condition);
```



Do while loops

```
int input;
do
{
    printf("Enter a positive integer: ");
    input = get_int();
}
while (input < 1);
```



Conditionals

if
if…else
if…else if…else
switch
ternary operator

If

```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    printf("Give me an integer: ");
    int n = get_int();

    if (n > 0)
    {
        printf("You picked a positive number!\n");
    }
}
```

If…Else

```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    printf("Give me an integer: ");
    int n = get_int();

    if (n > 0)
    {
        printf("You picked a positive number!\n");
    }
    else
    {
        printf("You picked a non-positive number!\n");
    }
}
```

If…Else if

```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    printf("Give me an integer: ");
    int n = get_int();

    if (n > 0)
    {
        printf("You picked a positive number!\n");
    }
    else if (n < 0)
    {
        printf("You picked a negative number!\n");
    }
}
```

If…Else if…Else

```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    printf("Give me an integer: ");
    int n = get_int();

    if (n > 0)
    {
        printf("You picked a positive number!\n");
    }
    else if (n < 0)
    {
        printf("You picked a negative number!\n");
    }
    else
    {
        printf("You picked 0!\n");
    }
}
```

Example

```
if (grade >= 90)
{
    printf("You got an A.\n");
}
if (grade >= 80 && grade < 90)
{
    printf("You got a B.\n");
}
if (grade >= 70 && grade < 80)
{
    printf("You got a C.\n");
}
if (grade >= 60 && grade < 70)
{
    printf("You got a D.\n");
}
if (grade < 60)
{
    printf("You got an F.\n");
}
```

Example

```
if (grade >= 90)
{
    printf("You got an A.\n");
}
else if (grade >= 80)
{
    printf("You got a B.\n");
}
else if (grade >= 70)
{
    printf("You got a C.\n");
}
else if (grade >= 60)
{
    printf("You got a D.\n");
}
else
{
    printf("You got an F.\n");
}
```

Switch

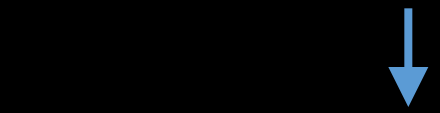
```
switch (input)
{
    case 0:
        printf("Your input was 0.\n");
        break;
    case 1:
        printf("Your input was 1.\n");
        break;
    case 5:
        printf("Your input was 5.\n");
        break;
    case -12:
        printf("Your input was -12.\n");
        break;
    default:
        printf("We can't handle that input.\n");
        break;
}
```

Ternary Operator

```
int n = get_int();
string str;
if (n > 100)
{
    str = “high”;
}
else
{
    str = “low”;
}
printf(“You picked a %s number.\n”, str);
```

Ternary Operator

```
int n = get_int();
string str;
if (n > 100)
{
    str = “high”;
}
else
{
    str = “low”;
}
printf(“You picked a %s number.\n”, str);
```



```
int n = GetInt();
string str = (n > 100) ? “high” : “low”;
printf(“You picked a %s number.\n”, str);
```

Numeric Data Types

	size (bytes)	maximum	minimum	precision
int	4 (on IDE)	2^31 - 1	-2^31	n/a
float	4	~10^38	~-10^38	6 decimal places of precision
double	8	~10^308	~-10^308	15 decimal places
long long	8	2^63-1	-2^63	n/a
unsigned int	4	2^32-1	0	n/a
char	1	127	-128	n/a

Floats

```
float imprecise = 1.0 / 10.0;
printf("%.20f\n", imprecise);
// prints 0.10000000149011611938
```

Floats have 4 parts:
(sign) precision * (base ^ exponent)
So as a float, the value 1.2345 = (+) 12345 * (10 ^ -4)
As the exponent grows, less space to store precision

Numeric Data Types

	size (bytes)	maximum	minimum	precision
int	4 (on IDE)	2^31 - 1	-2^31	n/a
float	4	~10^38	~-10^38	6 decimal places of precision
double	8	~10^308	~-10^308	15 decimal places
long long	8	2^63-1	-2^63	n/a
unsigned int	4	2^32-1	0	n/a
char	1	255	0	n/a

Chars

```
printf(“%d\n”, ‘A’ + 1) // prints 66
printf(“%c\n”, 65 + (‘a’ – ‘A”) // prints a
```

Chars are literally ints

‘A’ = 65

‘a’ = 97

Two’s complement

- we know how to represent positive numbers in binary
- we know how to add and subtract
- no idea how to represent negatives

Two’s complement

-1 should be something that will sum with 1 to get 0

```
  00000001
+        x
----------
  00000000
```

Two’s complement

-1 should be something that will sum with 1 to get 0

```
  00000001
+ 11111111
----------
 100000000
```

Two’s complement

-5 should be something that will sum with 5 to get 0

```
  00000101
+ 11111011
----------
 100000000
```

Two’s complement

-5 should be something that will sum with 5 to get 0

```
  00000101
+ 11111011
----------
 100000000
```

to get the complement of an int, flip bits and add 1

Two’s complement

- arithmetic operations work
- we don’t have to represent +/- 0
- decreases MAX_INT from 2^32 - 1 to 2^31 - 1