Nicole Luong

Week 3

While loops, Structs, Enums, Variable Names

Before we begin

Help Sessions

Lab check-ins

While Loops

Hand execute

```
#include <stdio.h>

int main(void) {

    int count = 100;
    while (count < 300) {
        printf("%d\n", count);
        count += 100;
    }

    return 0;
}
```

While loops

Hand execute these loops

A

```c
void a(void) {
    int i = 5;
    while (i > 0) {
        printf("%d\n", i);
        i--;
    }
}
```

B

```c
void b(void) {
    int i = 1;
    while (i < 32) {
        printf("%d\n", i);
        i = i + i;
    }
}
```

C

```c
void c(void) {
    int i = 0;
    while (i < 32) {
        printf("%d\n", i);
        i = i + 2;
    }
}
```

D

```c
void d(void) {
    int i = 5;
    while (i >= 0) {
        printf("%d\n", i);
        i--;
    }
}
```

E

```c
void e(void) {
    int i = 0;
    int keep_going = 1;
    while (keep_going == 1) {
        if (i > 3) {
            keep_going = 0;
        }
        i++;
    }
    printf("%d\n", i);
}
```

F

```c
void f(void) {
    int i;
    while (i > 0) {
        printf("%d\n", i);
        i--;
    }

}
```

G

```c
void g(void) {
    int i = 0;
    int max = 32;
    while (i < max) {
        printf("%d\n", i);
        max = max + 2;
    }

}
```

H

```c
void h(void) {
    int i = 0;
    int keep_going = 0;
    while (keep_going == 1) {
        if (i > 3) {
            keep_going = 0;
        }
        i++;
    }
    printf("%d\n", i);
}
```

2D While loops

Draw these 4 grids and fill out the patterns printed out by these loops


```c
void a(void) {
    int row = 0;
    while (row < SIZE) {
        int col = 0;
        while (col < SIZE) {
            if (row == col) {
                printf("O");
            } else {
                printf("X");

            }
            col++;
        }
        row++;
        printf("\n");
    }
}
```

```c
void b(void) {
    int row = 0;
    while (row < SIZE) {
        int col = 0;
        while (col < SIZE) {
            if (col % 2 == 0) {
                printf("O");
            } else {
                printf("X");

            }
            col++;
        }
        row++;
        printf("\n");
    }
}
```

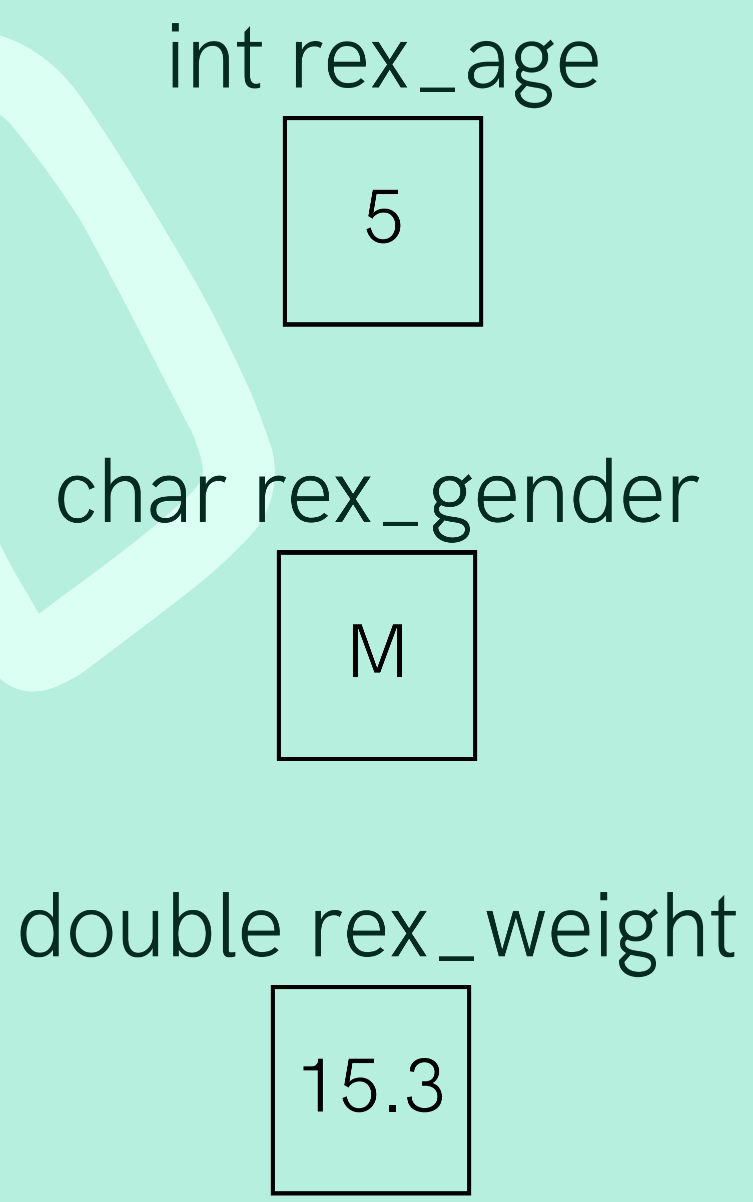
```c
void c(void) {
    int row = 0;
    while (row < SIZE) {
        int col = 0;
        while (col < SIZE) {
            if (col != 1 && row != 1) {
                printf("O");
            } else {
                printf("X");

            }
            col++;
        }
        row++;
        printf("\n");
    }
}
```

```c
void d(void) {
    int row = 0;
    while (row < SIZE) {
        printf("X");
        int col = 1;
        while (col < 3) {
            if (row == 0 || row == 3) {
                printf("X");
            } else {
                printf("O");
            }
            col++;
        }
        printf("X");
        row++;
        printf("\n");
    }
}
```

Structs

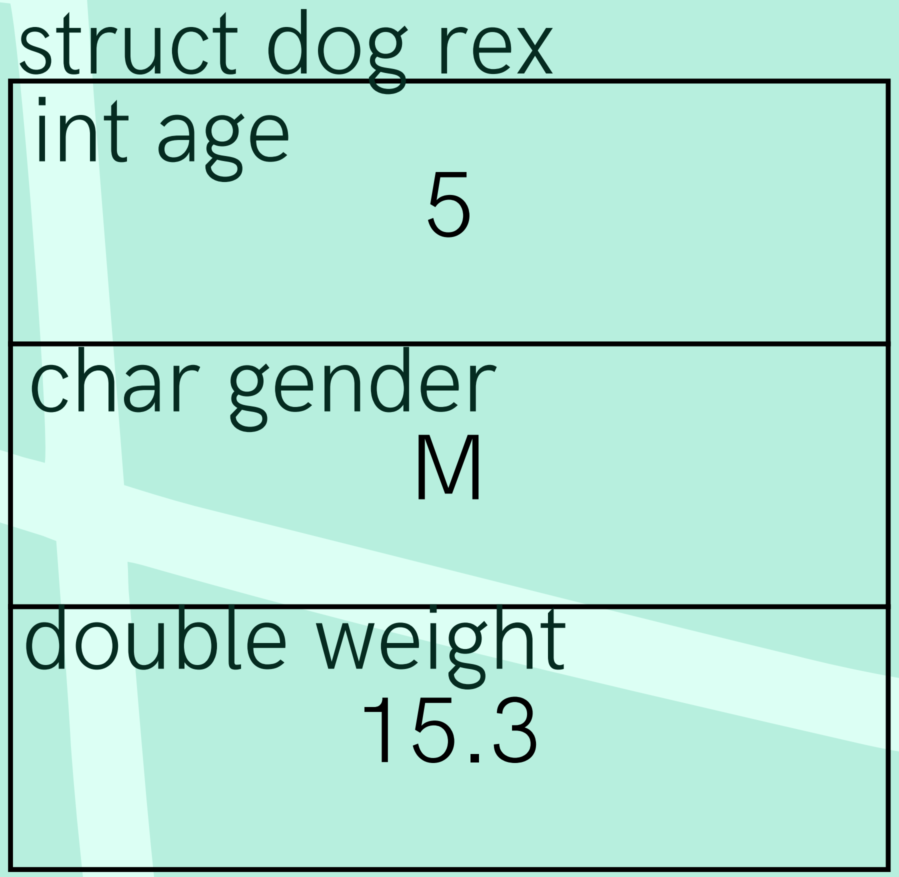
User defined datatype that allows users to group together items of different types

Instead of this:



Do this:

```
struct dog {
  int age;
  char gender;
  double weight;
}
```



Why?

- Store related information in a single datatype rather than multiple
- Can return more information from functions

Enums

User defined datatype that allows users to assign names to predefined constants

Instead of this:

```
#define MONDAY 0
#define TUESDAY 1
#define WEDNESDAY 2
#define THURSDAY 3
#define FRIDAY 4
#define SATURDAY 5
#define SUNDAY 6
```

Do this:

```
enum weekdays {
  MONDAY,
  TUESDAY,
  WEDNESDAY,
  THURSDAY,
  FRIDAY,
  SATURDAY,
  SUNDAY
};
```

Why?

- The type conveys more information to the user
  eg. int vs enum weekdays
- Assigns the constants for you

Variable Names

Legal Variable Names in C

- Contains letters, numbers, or _
- Must not start with a number

Good Style

- Start with lowercase
- Snake case eg. good_example_variable_name
- #defines names and enums must be in SHOUTING_SNAKE_CASE
- Descriptive and relevant to the program

Lab Time!