# ATTENDANCE!!!!

forms.office.com/r/**rJ7bpQUjq8**



**Author: Alan Chu (ualch9@gmail.com)**

# Today

- Do some PA stuff
- Do lab task 1
- Do some more exam review
- Do lab task 2
- Do some more exam stuff
- Go home and cry (as a team or individually)


**Author: Alan Chu (ualch9@gmail.com)**

# Where should I start?

- Use Andy's functions for an outline, like an essay.
- Add additional functions as needed.
- Don't worry about how the functions work, as long as you are concerned, these functions do as they say.

- `welcome_screen`
- `initialize_game_board`
- `select_who_starts_first`
- `manually_place_ships_on_board`
- `randomly_place_ships_on_board`
- `check_shot`
- `is_winner`
- `update_board`
- `display_board`
- `output_current_move`
- `check_if_sunk_ship`
- `output_stats`
- `That's all, unless…?`


**Author: Alan Chu (ualch9@gmail.com)**
https://github.com/ualch9/cpts121-lab-presentations

# Example Outline/Floatchart

1. initialize_game_board(board)
2. select_who_starts_first() [store in currentPlayer]
3. ask_user_random_or_manual_placement()
   - If 0: randomly_place_ships_on_board(playerOneBoard)
   - If 1: manually_place_ships_on_board(playerOneBoard)
4. get_coordinates_to_attack()
5. etc…


**Author: Alan Chu (ualch9@gmail.com)**
https://github.com/ualch9/cpts121-lab-presentations

# Task 1: RNG To Make Sentences

- Four arrays: **`article`**, **`noun`**, **`verb`**, **`preposition`**
- Select a word at random in such order:
  - article, noun, verb, preposition, article, noun.
- Words are added to a sentence array
- When outputting the sentence, it starts with a capital letter and ends with an exclamation mark

- The program generates 20 sentences

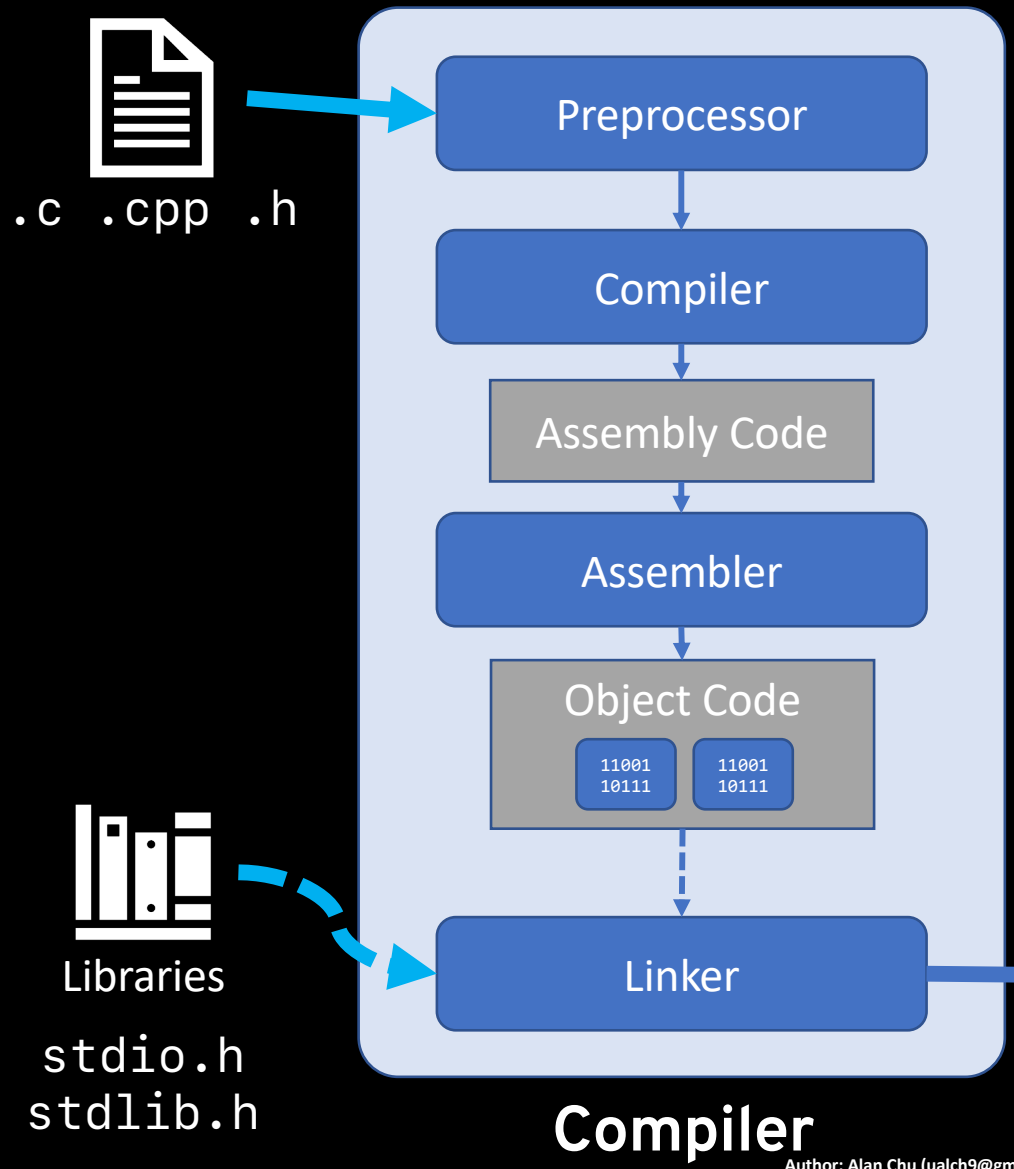

**Author: Alan Chu (ualch9@gmail.com)**

# Review part 2

Let’s talk structs and compilers, eh?


**Author: Alan Chu (ualch9@gmail.com)**

**this is not on exam 2**




**PREPROCESSOR**

- Removes comments
- Insert .h files into the c/cpp file itself
- Replace macros
- etc.

**COMPILER**

- Generates assembly code

**ASSEMBLER**

- Generates object code



**Author: Alan Chu (ualch9@gmail.com)**

# Task 2: Tic Tac Toe

- The gameboard is represented as a 2D array

```c
typedef struct coordinate {
    int row;
    int col;
} Coordinate;
```

```c
typedef struct cell {
    int occupied;
    char symbol;
    Coordinate location;
} Cell;
```

- The gameboard size is chosen by the user
- But, you have to specify an array size for the compiler
- We aren't talking about dynamically allocated memory yet.


**Author: Alan Chu (ualch9@gmail.com)**
https://github.com/ualch9/cpts121-lab-presentations

# Reasonable Limits

- The compiler must know the size of the array

```
char foo[];
```

🛑 `Definition of variable with array type needs an explicit size or an initializer.`

```
char foo[64];
```

👍

We reasonably expect that the user's input will be within 63 characters long (-1 for NULL char).

```
char name[] = "Ashley";
```

👍

The compiler can infer that `name` will be 7 characters (+1 for NULL) long.


**Author: Alan Chu (ualch9@gmail.com)**
https://github.com/ualch9/cpts121-lab-presentations

# Reasonable Limits

```
int size;
scanf("%d", &size);

int matrix[size][size];
```

- expression must have constant value
- cannot allocate an array of constant size 0
- ’matrix’: unknown size

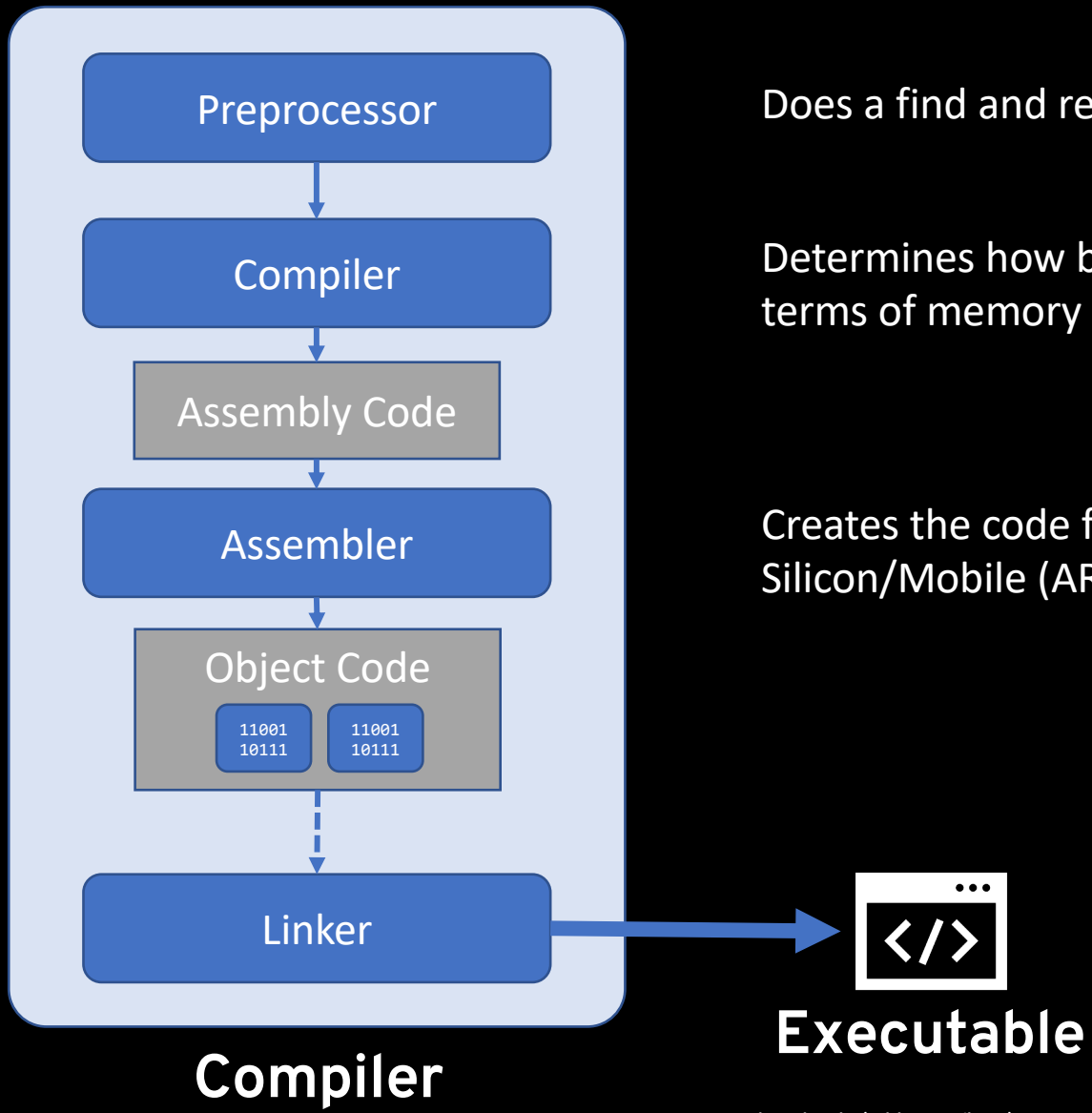

**Author: Alan Chu (ualch9@gmail.com)**

# Reasonable Limits

```c
#define kMAX_DIMENSION 10

char board[kMAX_DIMENSION][kMAX_DIMENSION];


// Initialize the matrix
for (int row = 0; row < kMAX_DIMENSION; row++) {
    for (int col = 0; col < kMAX_DIMENSION; col++) {
        board[row][col] = '\0';
    }
}


// Input validation loop
int boardSize = 0;
do {
    printf("> Please enter board size [between 3 and %d]: ", kMAX_DIMENSION);
    scanf("%d", &boardSize);
} while (boardSize < 3 || boardSize > kMAX_DIMENSION);
```


**Author: Alan Chu (ualch9@gmail.com)**
https://github.com/ualch9/cpts121-lab-presentations

**this is not on exam 2**




Does a find and replace of `kMAX_DIMENSION`

Determines how big to make the matrix (in terms of memory and pointers)

Creates the code for Intel/AMD (x86) or Apple Silicon/Mobile (ARM).



**Author: Alan Chu (ualch9@gmail.com)**

## this is not on exam 2

```
main: # @main
      pushq %rbx
      subq $16, %rsp
      movl $0, 12(%rsp)
      leaq 12(%rsp), %rbx
.LBB0_1:
      movl $.L.str, %edi
      movl $10, %esi
      xorl %eax, %eax
      callq printf
      movl $.L.str.1, %edi
      xorl %eax, %eax
      movq %rbx, %rsi
      callq __isoc99_scanf
      movl 12(%rsp), %eax
      addl $-3, %eax
      cmpl $7, %eax
      ja .LBB0_1
      xorl %eax, %eax
      addq $16, %rsp
      popq %rbx
      retq

.L.str:
      .asciz "> Please enter board size [between 3 and %d]: "

.L.str.1:
      .asciz "%d"
```



**Author: Alan Chu (ualch9@gmail.com)**
https://github.com/ualch9/cpts121-lab-presentations

# Task 2: Tic Tac Toe

- The gameboard is represented as a 2D array

```c
typedef struct coordinate {
    int row;
    int col;
} Coordinate;
```

```c
typedef struct cell {
    int occupied;
    char symbol;
    Coordinate location;
} Cell;
```

```c
Cell board[kMAX_DIMENSION][kMAX_DIMENSION];
```

```c
board[row][col].symbol = 'X';
```

'O' means occupied by o
'X' means occupied by x
'–' means unoccupied


**Author: Alan Chu (ualch9@gmail.com)**
https://github.com/ualch9/cpts121-lab-presentations