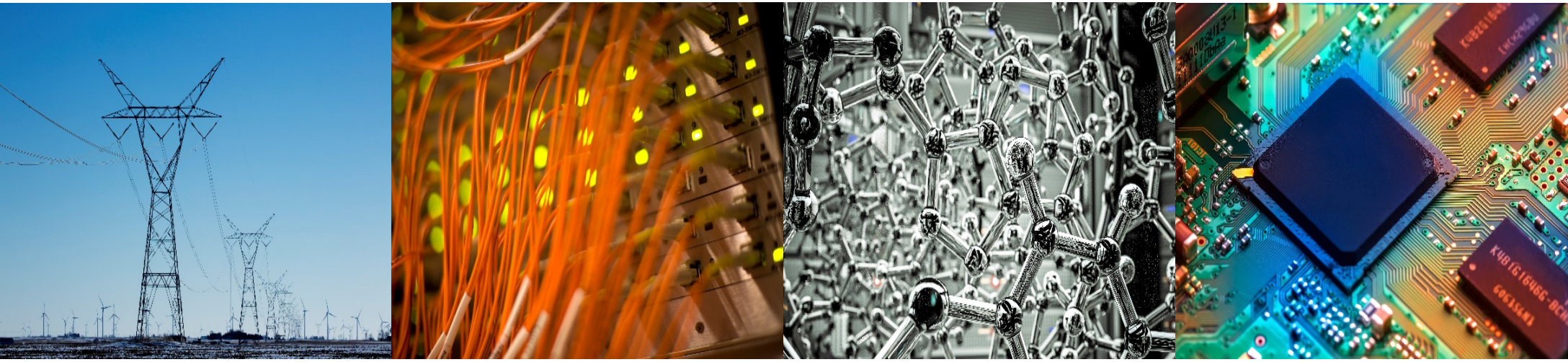
# ECE 220 Computer Systems & Programming

## Lecture 8 – Introduction to C: Control Structures

## Feb 18, 2021



- **MP2 is due Friday at 10pm CT**
- **MT1 practice questions are posted**

ILLINOIS
Electrical & Computer Engineering
GRAINGER COLLEGE OF ENGINEERING

# Basic I/O

```
#include <stdio.h>          /* header file for Standard Input Output */
```

- **printf**
  **(print to screen)**

```
printf("%d is a prime number", 43);
printf("43 + 59 in decimal is %d\n", 43+59);
printf("a+b=%f\n", a+b);
printf("%d+%d=%d\n", a, b, a+b);
```

- **scanf**
  **(get user input)**

```
scanf("%c", &nextchar);
scanf("%f", &radius);
scanf(%d %d", &length, &height);
```

Formatting option: %d, %x, %c, %s, %f, %lf, \n,

Use "**man**" to look up library functions

https://play.kahoot.it/v2/?quizId=611eb370-7016-4470-bf42-abb295fb1857

# Lecture 7 Review

- **global vs. local (scope)**

```
int global_x = 5;
int main(){
   int local_x = 10;
   printf("global=%d, local=%d\n", global_x, local_x);
   {
      int local_x = 15;
      global_x = 20;
   }
   printf("global=%d, local=%d\n", global_x, local_x);
   return 0;
}
```

- **const vs. static (qualifier)**

```
const int x = 0;
static int y = 0;
```

- **' = ' vs. ' == '**

```
int x = 1, y = 2;
int result = (x == y);
```

- **pre vs. post decrement**

```
int x = 4; int y = --x;
int x = 4; int y = x--;
```

# Control Structures

Conditional Constructs

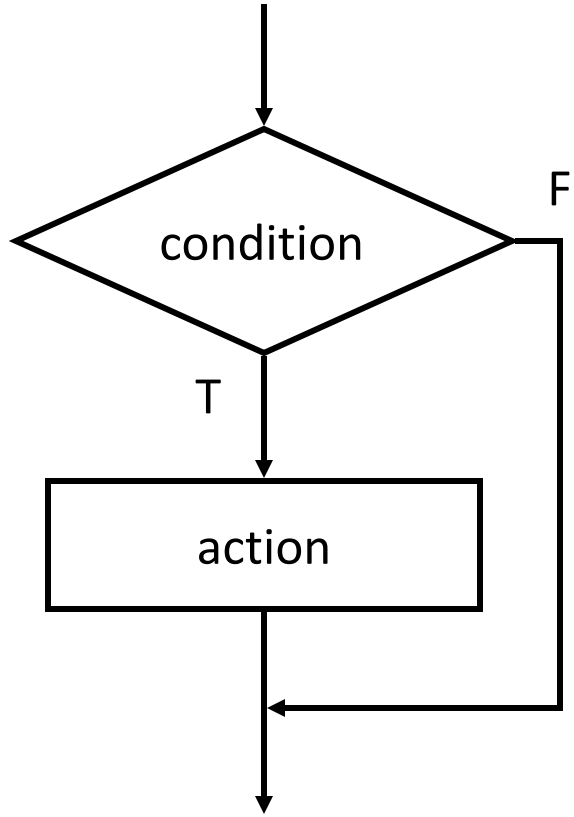
- if
- if - else
- switch

Iteration Constructs (loops)

- while
- do - while
- for

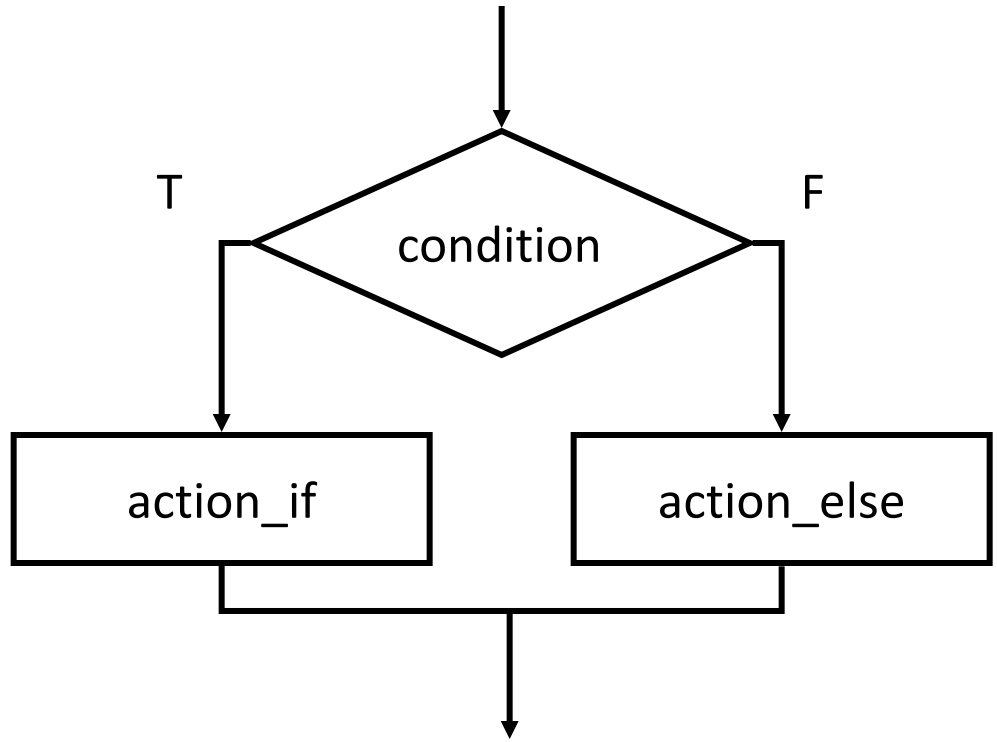
# The if Statement (similar to BR in LC-3)

```
int x, y;
/* assign some value to x, code omitted */

if (x < 0)
   x = -x; /* invert x only if x < 0 */


if ((x > 10) && (x < 20)){
   y = x % 3;
   /* What would be the value of y? */
   printf("y = %d\n", y);
}
```



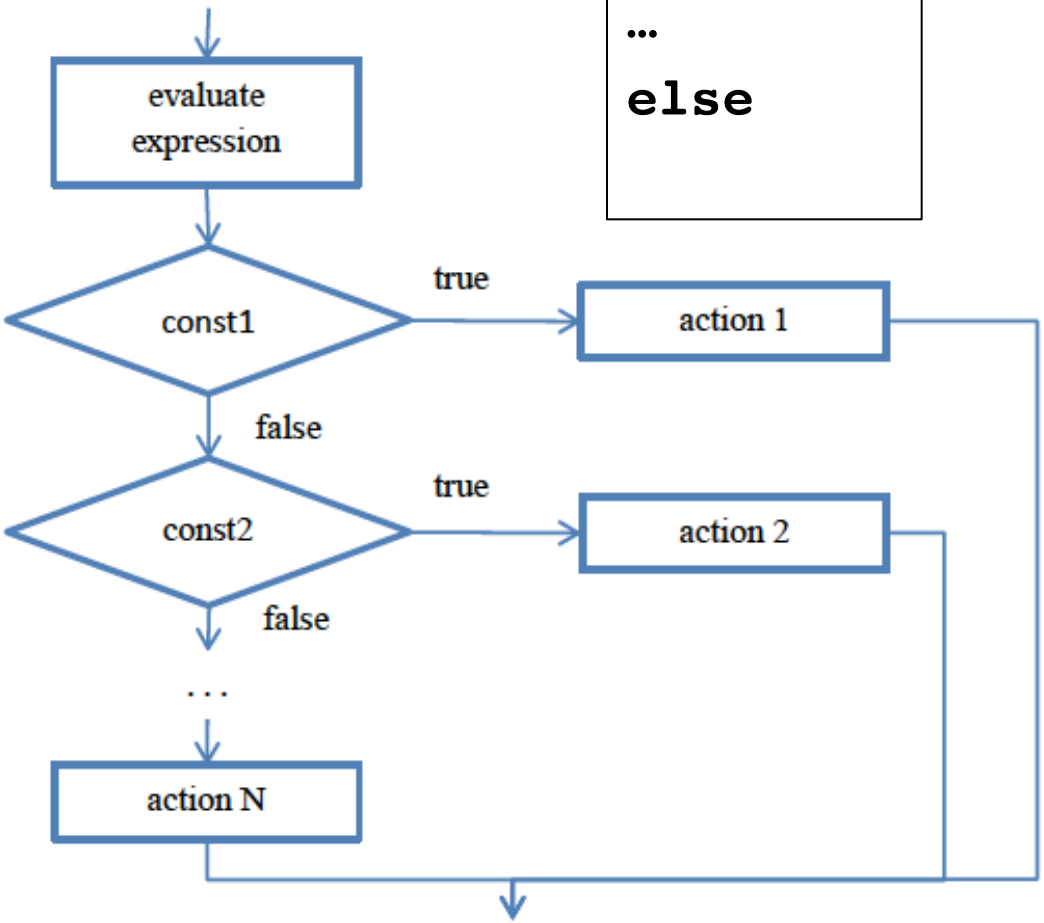
# The if - else Statement

```
int x, y;
/* assign some value to x,
code omitted */

if (x < 0)
   x = -x;
else
   x = x * 2;

if ((x > 10) && (x < 20)){
   y = x % 3;
   printf("y = %d\n", y);
}
else
   printf("x = %d\n", x);
```



# The switch Statement

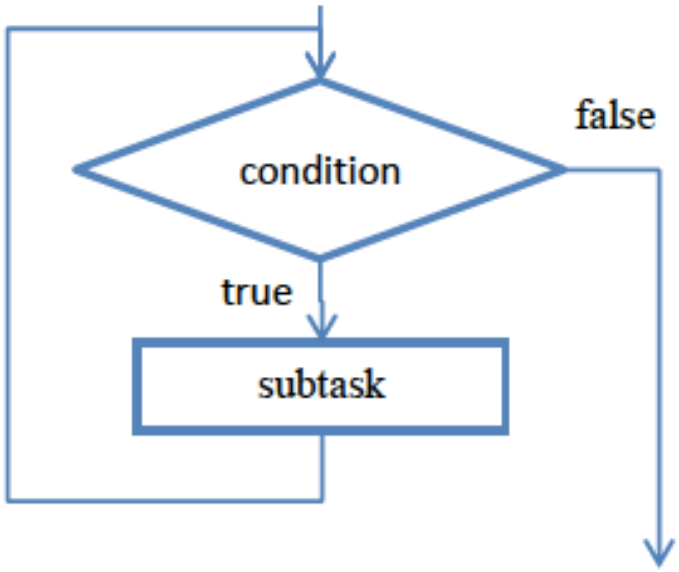
```
if
else if
else if
…
else
```




```
switch (expression){
    case const 1:
            action 1;
            break;
    case const 2:
            action 2;
            break;
    ...
    default:
            default action;
            break;
}

/*notice the use of 'break'*/
```

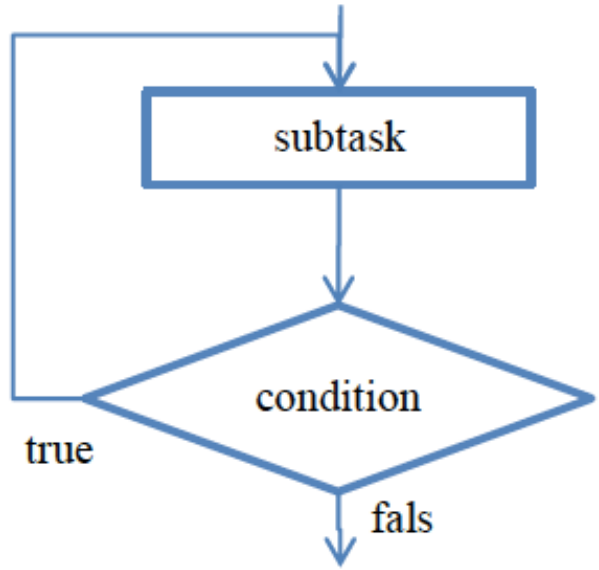
# The while / do - while Statement

while: loop body may or may not be executed even once



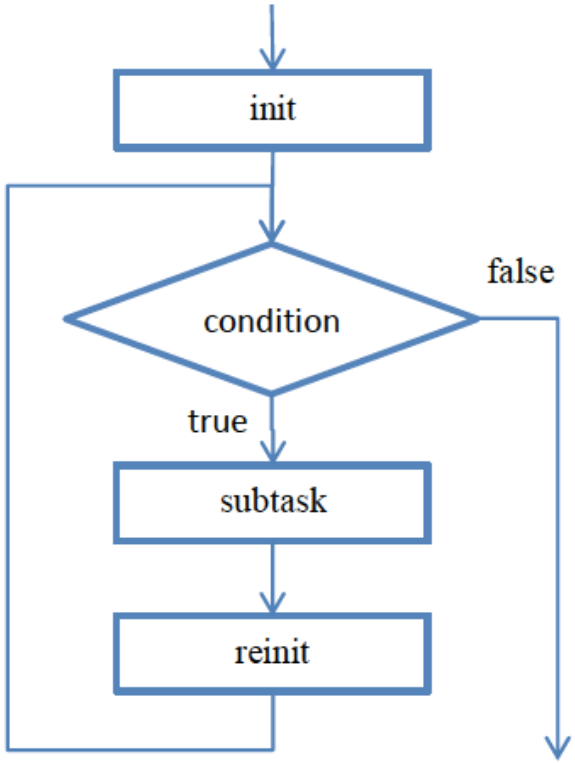

```
int x = 0;
while (x < 10) {
    printf("x=%d\n", x);
    x = x + 1;
}
```

do – while: loop body will be executed **at least once**




```
int x = 0;
do {
    printf("x=%d\n", x);
    x = x + 1;
} while (x < 10);
```

# The for Statement




```
int x = 0;
while (x < 10) {
    printf("x=%d\n", x);
    x = x + 1;
}
```

```
int x;
for (x = 0; x < 10; x++){
   printf("x=%d\n", x);
}
```

- What could cause while loop or for loop to become infinite loops?

```
for (x = 0; x < 10; x++){
    if (x == 5)
        break;
    printf("x=%d\n", x);
} /* what would be the output? What if
'break' is replaced with 'continue'? */
```

# Nested Loops

inner loop is nested within the outer loop (similar to print hex example in LC-3)

```
for (){
   for (){
      …
   } /* inner loop to shift 4 bits to calculate each digit */
   …
} /* outer loop to print the 4 digits */
```

# Exercise

Write a program to print an n x n identity matrix using nested loops.

```
#include <stdio.h>
#define N 5
int main(){



}
```

# Follow-up Questions

- What are some ways to stop after printing the third '1' on the main diagonal, such as the example below?

  1 0 0 0 0

  0 1 0 0 0

  0 0 1

- How can we take user input for the value of n?

- How can we add a check before printing the matrix to ensure user input is within the valid range of 0 < n < 10? (If user input is invalid, print the message "Number entered is invalid" and prompt the user to enter a number again. )