**For loops:**

- for (expr1; expr2; expr3) {statements}

- e1 is an initializer before loop is executed

- e2 is usually a condition

-e3 is usually an increaser/decreaser

-in C99 and beyond we can initialize a variable in e1 that is only used in the loop

ex1 -> ex2 -> true -> statements -> ex3 -> if still true, back to ex 2, else ends for loop

1. for(int i = 0; i < 10; i = i+1){
2. printf("%d\n", i)
3. output: 0 1 2 3 ... 9
4. Note: using i outside of the loop now would result in an error

**Breaks and Continues:**

- break will terminate a while, do, or for loop

- C also has continue statements to go to just before the end of loop body

- will skip the rest of the loop, check again if statement is true

- in for loops, will skip the rest, perform expr3, and check again if expr2 is satisfied

- goto command can jump to any labeled part of the program

- LIMIT these functions, use wisely as they can make code tough to debug and read

1. int n = 0, sum =0, i =0;
2. while (n <3) {
3. scanf("%d", i);
4. if (i==0) continue;
5. sum += i;
6. n++;
7. }
8. print sum;
9. }
10. int i is scanned, not used as 0 every time
11. we can also say if(i!=0){sum, n++}
12. this would allow us to avoid using continue in this case, cleaning up our code
13. only want to print once, which is why printf needs to be outside of the brackets

**Nested loops:**for (){

for(){

}}

- we can put loops inside of loops, as they are both valid statements

**asks user for int and checks if prime**

1. scanf("%d", int i);
2. int div = 2;
3. if(i <= 1){
4. printf("Not Prime");
5. }
6. else{
7. while (div\*div <= n){
8. if (n%div == 0) break;
9. div++;
10. }
11. if (div\*div <= n){ print ("Not Prime")}
12. else printf("Prime")

**Comments:**

- code not works but isnt very usable

- would be nice to create a function that would return true or false depending on the answer

- This is done through functions

1. we can use scanf to return the number of integers
2. //while read is successful
3. while(scanf("%d", d) == 1){  
   sum += d
4. }
5. printf("%d", sum)

**Functions:**