comp10002 Week 3 Workshop

loop-de-looping

Discussions	• Loops, ex. 4.02, 4.01 • Working with data type char • Sample Exercise: how to start 4.07 + I/O redirection
Lab	• Minimal: 4.05, 4.06, 4.07 • Additional: 5.06, or perhaps 4.09, 4.10, 4.11

B a C Pro

Compare:

a= 5;
b= 5;
	a= b= 5; // b=5 is also an expression!
	// assignments evaluated from right to left

a= a * b;
a= a+b;
n= n+1;
m= m-1;
	a *= b;
	a += b;
	n++; n += 1;
	m--;

scanf("%d%d", &a, &b);
//rest of the program
	if (scanf(%d%d", &a, &b) != 2) {
	  printf("invalid input\n");
	  exit(EXIT_FAILURE);
	}
	…

/* a loop for reading and processing a number of pairs (a,b) */	while (scanf("%d%d", &a, &b) == 2) {
	  // do something with the new value of a and b
	}

Loops: the while loop and the for loop

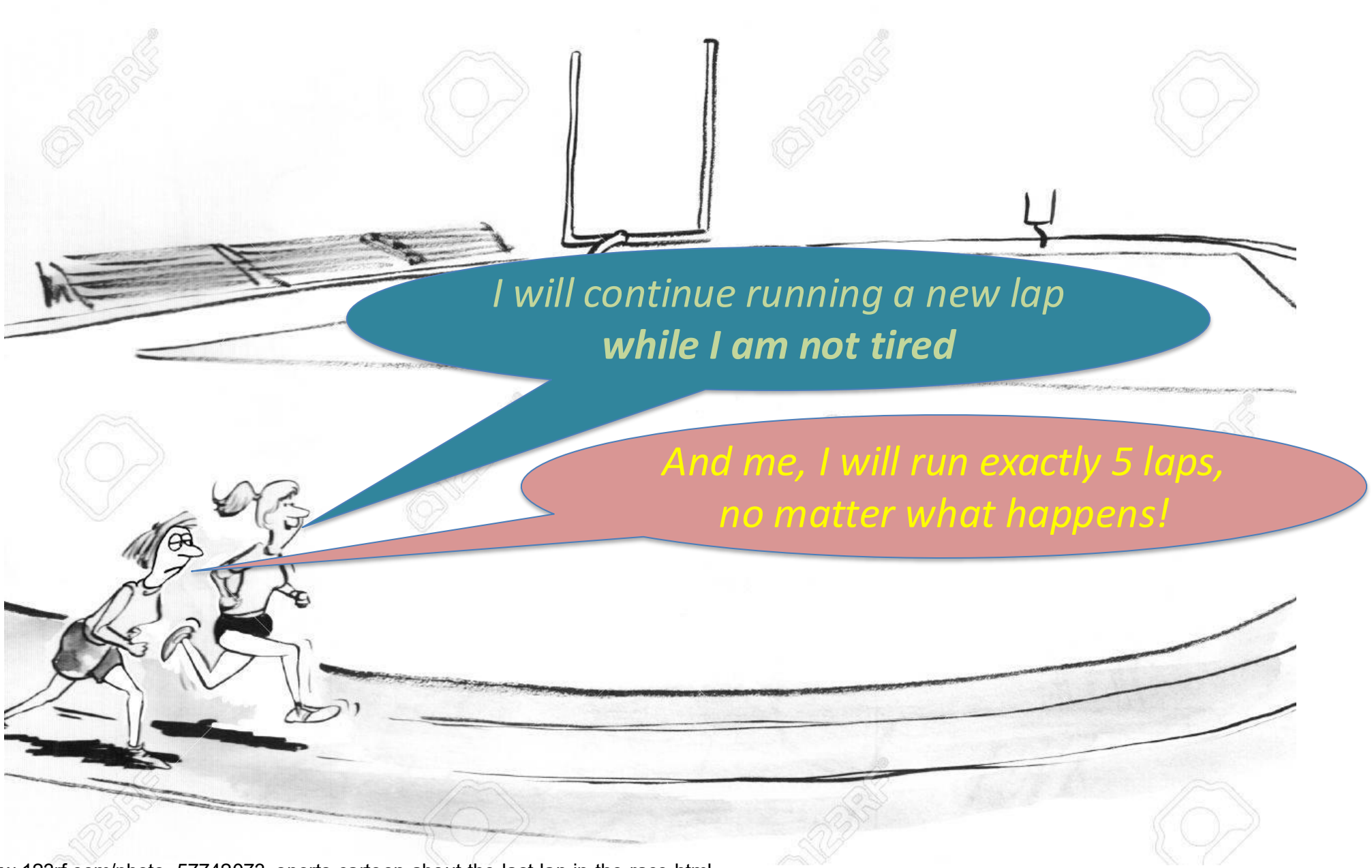


image: adapted from https://www.123rf.com/photo_57742073_sports-cartoon-about-the-last-lap-in-the-race.html

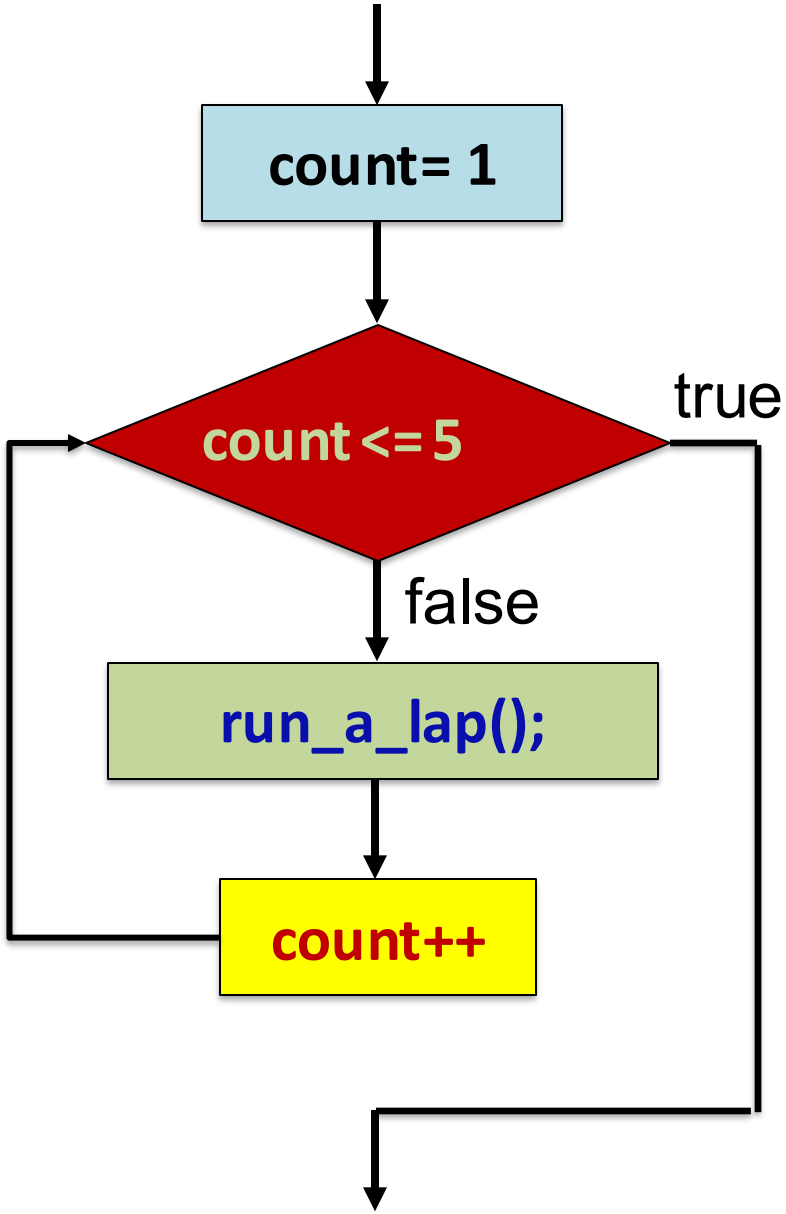
Running exactly 5 laps: the while loop vs. the for loop

```
count= 0
while ( count < 5 ) {
    run_a_lap();
    count++ ;
}
```

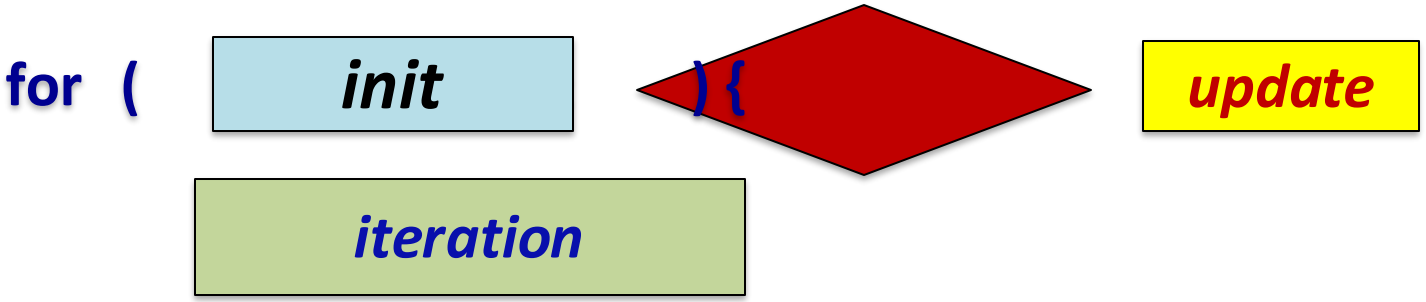
same as:

```
for ( count= 1 ; ) {
    run_a_lap();
}
```

init *guard* *update* *iteration*



the for loop



- all components of the **for** can be empty, the only compulsory part is **for (; ;)**
- empty *guard* means *1*

Question 1: Any other way to write the loop:

```
for (count= 1; count <= 5; count++) {
   run_a_lap();
}
```

Question 2

Write a C fragment to compute the sum of all perfect squares that are smaller than a given integer n.
That is, compute

$1^2 + 2^2 + 3^2 + \ldots + k^2$

where k is the largest value that satisfies $k^2 < n$

Another, but Dangerous, Loop: the `do…while` loop

```
while ( guard ) {
  sequence of statements
}
```

```
do {
  sequence of statements
} while (guard);
```

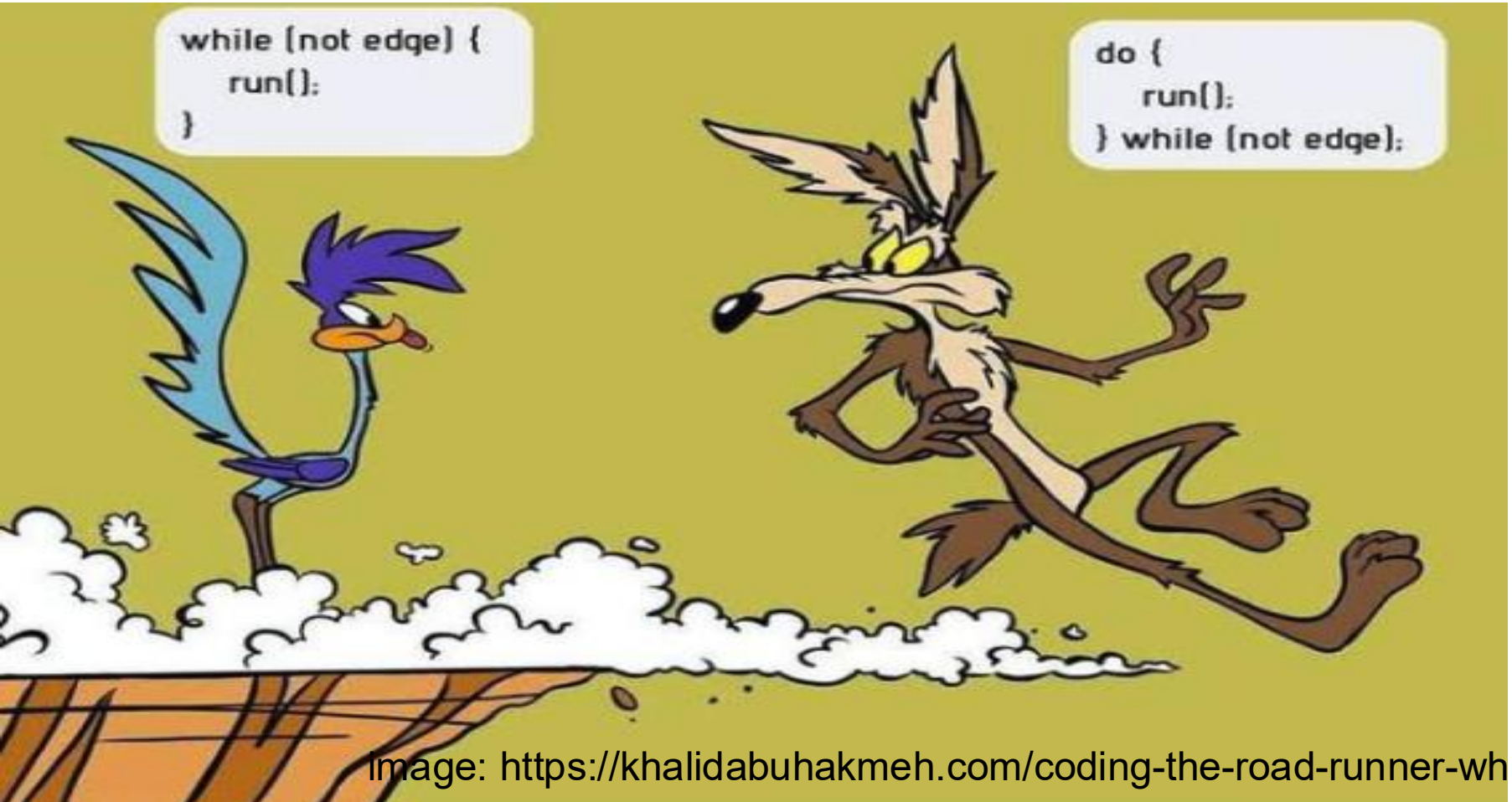


Image: https://khalidabuhakmeh.com/coding-the-road-runner-while-loop-meme

`exercise 4.02`

Give a general construction that shows how any do *statement can be converted into an equivalent* while *statement.*

Understanding the for loop: Exercise 4.1 on Ed. 4.1a

Sample question: Trace the action of the loop, and determine the values printed out by the printf statement. Assume that all variables have been declared to be of type int

```
1   for (i=0; i<20; i=i+3) {
2      printf("%2d\n", i);
3   }

    output = ?
```

Understanding the for loop: Exercise 4.1 on Ed. 4.1b

Sample question: Trace the action of the loop, and determine the values printed out by the printf statement. Assume that all variables have been declared to be of type int

```
1   for  (i=1; i<2000000; i= 2*i) {
2       printf ("%7d\n", i);
3   }

    output = ?
```

4.1c

```
1
2
3
4
5
6
```

```
sum = 0;
for  (i=1; i<10; i++) {
    sum = sum + i;
    printf (“S(%2d) = %2d\n”, i, sum);
}

i == ?
```

4.1 f

```
1   j = 5;
2   for (i= 0; i < j; i++);
3   {
4       printf (“i= %d, j= %d\n”, i, j);
5   }

    i=  , j=
```

4.1g

```
1    j = 5;
2    for (i= 0; i < j; j++) {
3        printf (“i= %d, j= %d\n”, i, j);
4    }
5
```

4.1d (*modified – not the same as in Ed*)

```
1   for (i= 0; i < 2; i++) {
2     for  (j= i+1; j < 8; j += 3) {
3       printf ("i= %d, j= %d\n", i, j);
4       // we are in the inner loop
5     }
6     // we are in the outer loop
7   }
8   ...
```

4.1e (*modified*)

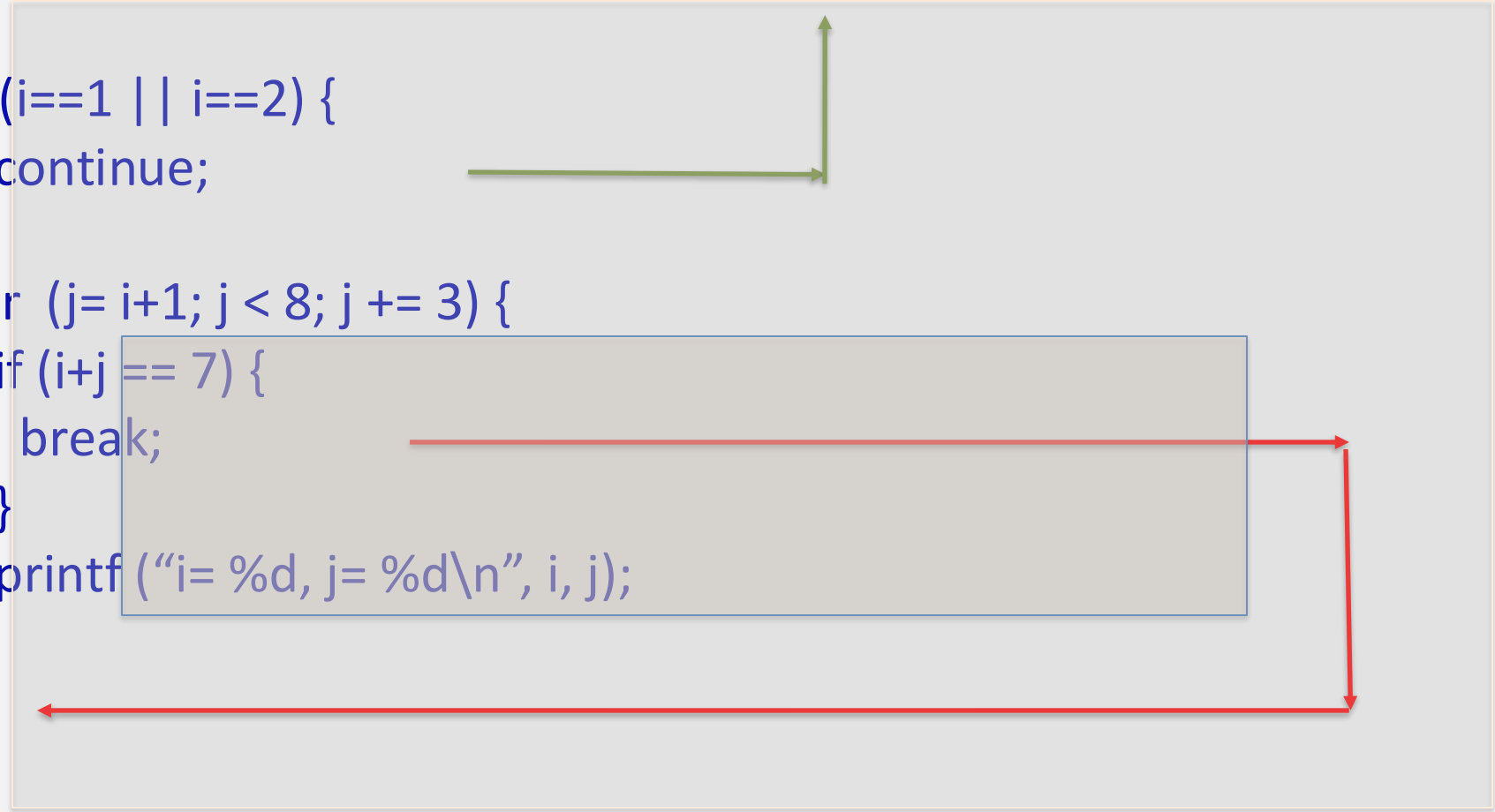
```
1   for (i= 0; i < 4; i++) {

2     if (i==1 || i==2) {
3       continue;
4     }
5     for  (j= i+1; j < 8; j += 3) {
6       if (i+j == 7) {
7         break;
8       }
9       printf (“i= %d, j= %d\n”, i, j);
10    }
11
12
13  }
14  ...
```

4.1e (*modified*)

```
1   for (i= 0 ;   i < 4 ;    i++) {

2     if (i==1 || i==2) {
3       continue;
4     }
5     for  (j= i+1; j < 8; j += 3) {
6       if (i+j == 7) {
7         break;
8       }
9       printf (“i= %d, j= %d\n”, i, j);
10    }
11
12
13  }
14  …
```



Tell your mates: “My answer is ... You know I'm spittin' facts, right?” ☺

Quiz 1

How many lines and numbers are printed by the following segment:

```
int i,j;
for (i=0; i<10; i++) {
  if ( i % 2 ) continue;
  for (j=0; j<3; j++) {
    printf("%d ", i*j);
  }
  printf("\n");
  if (i==2) break;
}
```

A	9 lines, 9 numbers
B	2 lines, 6 numbers
C	3 lines, 9 numbers
D	4 lines, 12 numbers
E	none of above, or syntax error

Quiz 2

Supposing that all variables are pre-declared as **int**.

Which one correctly gets

s= $1^2 + 2^2 + \ldots + n^2$?

A	`while (i <= n) {`
	`  s += i*i;`
	`  i++;`
	`}`

B	`for (i=1; i<=n; i++) s += i*i;`
C	`for (s=0; n > 0; n--) s += n*n;`
D	`for (i=0, s=0; i<n; i++) s = s + i*i;`
E	`for (i=1, s=0; i<=n; i++) s += i**2;`

4.05 – Design (Discussion, view exercise in `Ed, Ex4.05`)

Design and implement a program (say, `grapher.c`) that reads integers and draw a simple graph. Assume that all of the values read are between 1 and 70. Example:

```
bash$ ./grapher
Enter integers between 1 and 70 inclusive: 3 7 11
  3 |***
  7 |*******
 11 |***********
```

4.05: Design – Implement – Run – Mark

Design and implement a program that reads integers and draw a simple graph. Assume that all of the values read are between 1 and 70. Example:

```
bash$ ./program
Enter integers between 1 and 70 inclusive: 3 11
  3 |***
 11 |***********
```

4.07 – Design & Implementation (Ed → Chapter4 → Ex4.07)

Design a program my_wc *that count the number of characters, words, and lines in the input. Example of execution:*

```
bash$ ./my_wc
Enter text:
Mary has a      little lamb
Little lamb, 1+2=3 little          lamb;
^D    (or ^Z if using MinGW/Windows)

Lines: 2
Words: = 9
Chars: ??
```

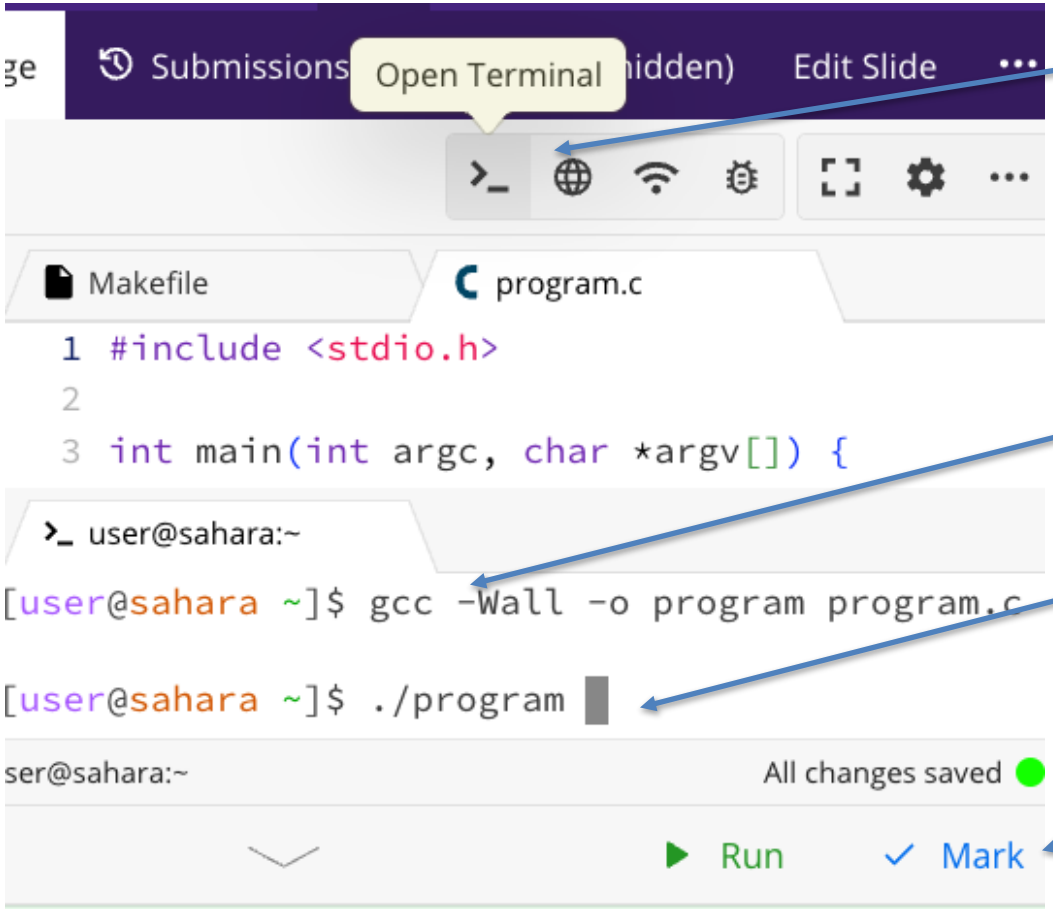
Understand the problem first! Make clear:

- What is a character, how to get it?
- What is a line, how to recognize it?
- What is a word, how to recognize it?

Also take notes on:

- ***datatype* char** and related tools/functions: %c, %d, scanf, getchar, isspace, isupper...
- ***I/O redirection*** and their use in saving time when testing/debugging

2nd Method to Run Code on Ed (instead of Clicking ▶ Run). Do 4.06, 4.07, 4.04



First, click >_ to open Terminal

Then, on the Terminal:

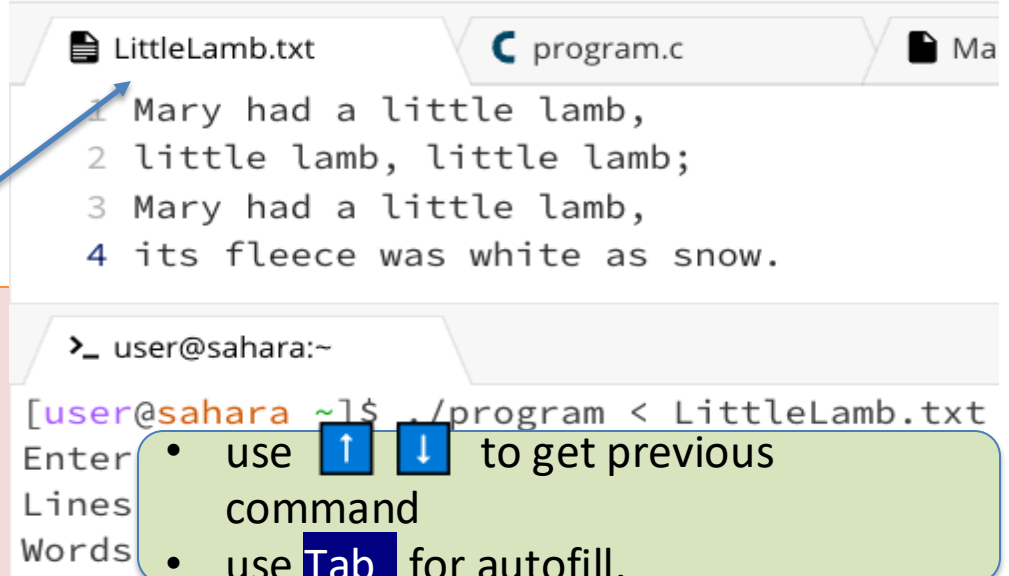
1. Compile program.c to executable program:

```
$ gcc -Wall -o program program.c
```

2. Run the executable file

```
$ ./program
```

Remember to click ✓ Mark



The main Advantage of the 2nd method is the Flexibility, for example:

- having more than just one .c files in the workspace
- run executable file with file redirection, e.g. (Ex4.07):

```
$ ./program < LittleLamb.txt
```

Lab and Wrap-Up

Minimal Implementation:

- Exercise 4.5: Simple character graph
- Exercise 4.7: Counting characters, words, and lines

Extra Implementation in grok

- Exercise 4.9: Computing the next prime number
- Exercise 4.4: Printing (a part of) the ASCII table
- Exercise 4.3: Computing Fibonacci numbers
- Exercise 4.10,11 [hard]: Replace C99 comments

Wrap-Up, today's topics:

- the for loop is general & powerful
- the while loop is similar to that in Python
- char vs int, scanf("%c")/getchar(), functions in <ctype.h>
- using file redirection