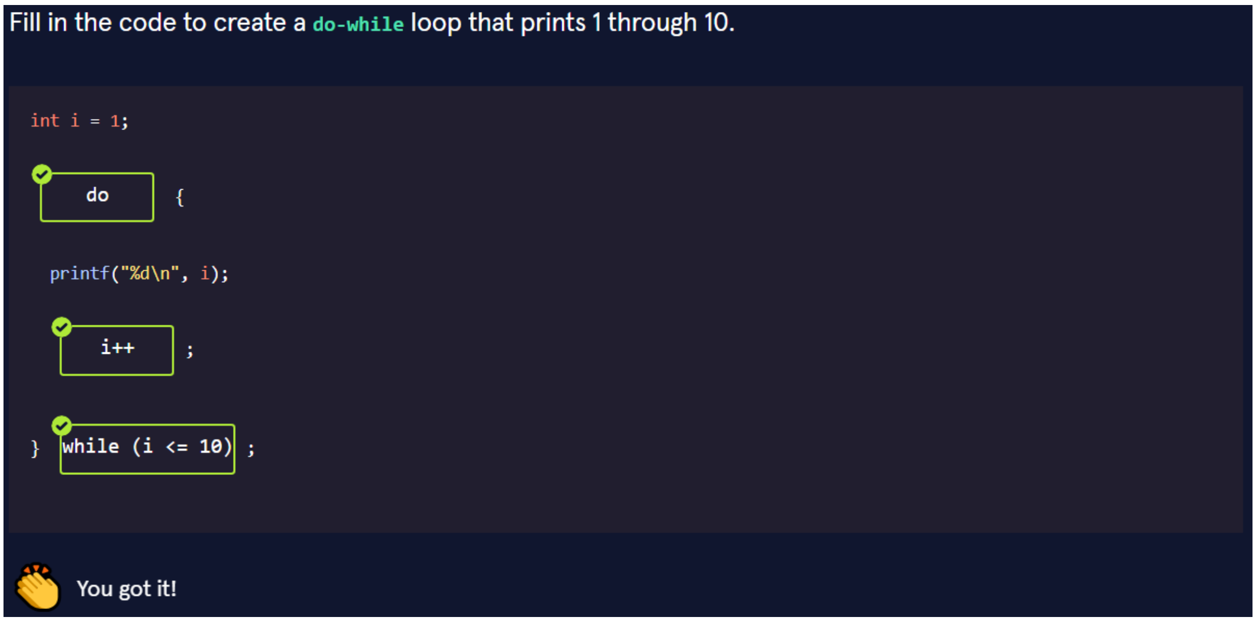
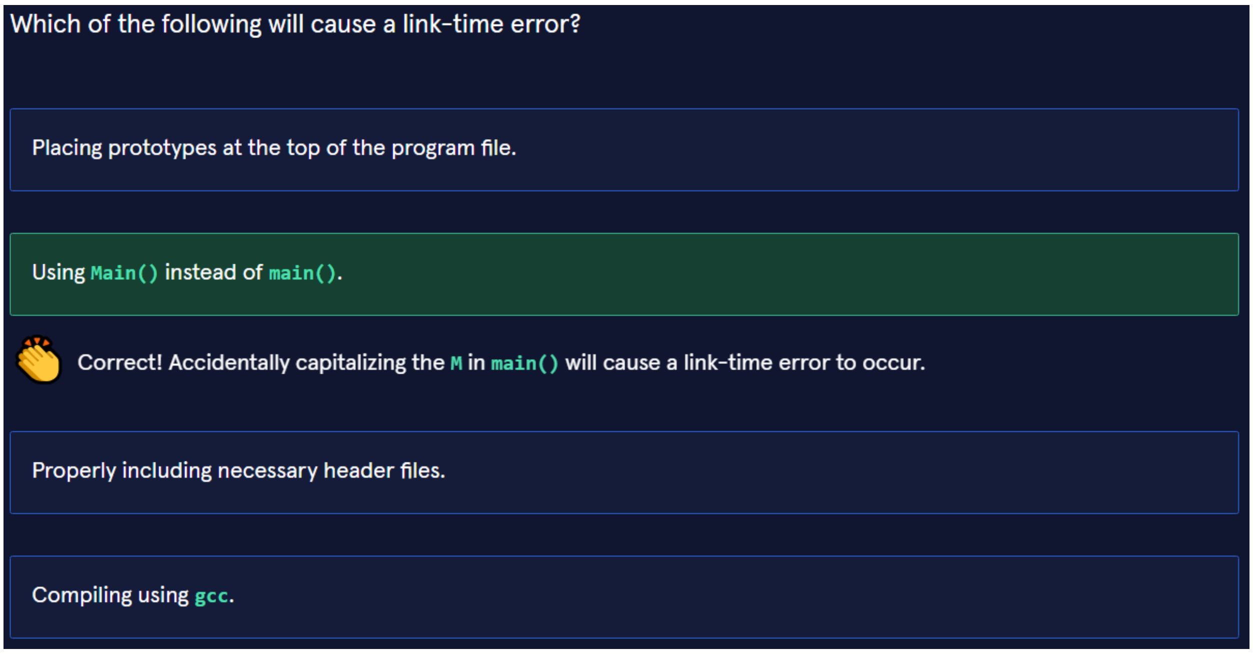
QUIZ


Fill in the code to create a `do-while` loop that prints 1 through 10.

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

👏 You got it!


Which of the following will cause a link-time error?

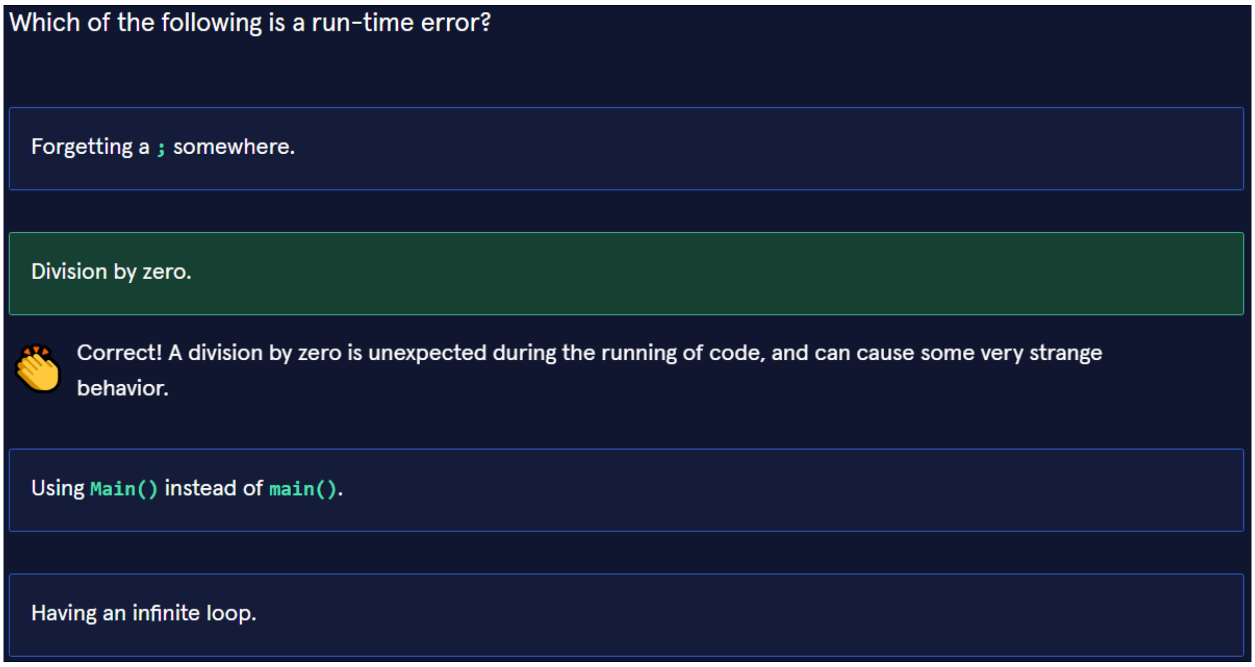
Placing prototypes at the top of the program file.

[✓ Using `Main()` instead of `main()`.]

👏 Correct! Accidentally capitalizing the `M` in `main()` will cause a link-time error to occur.

Properly including necessary header files.

Compiling using `gcc`.

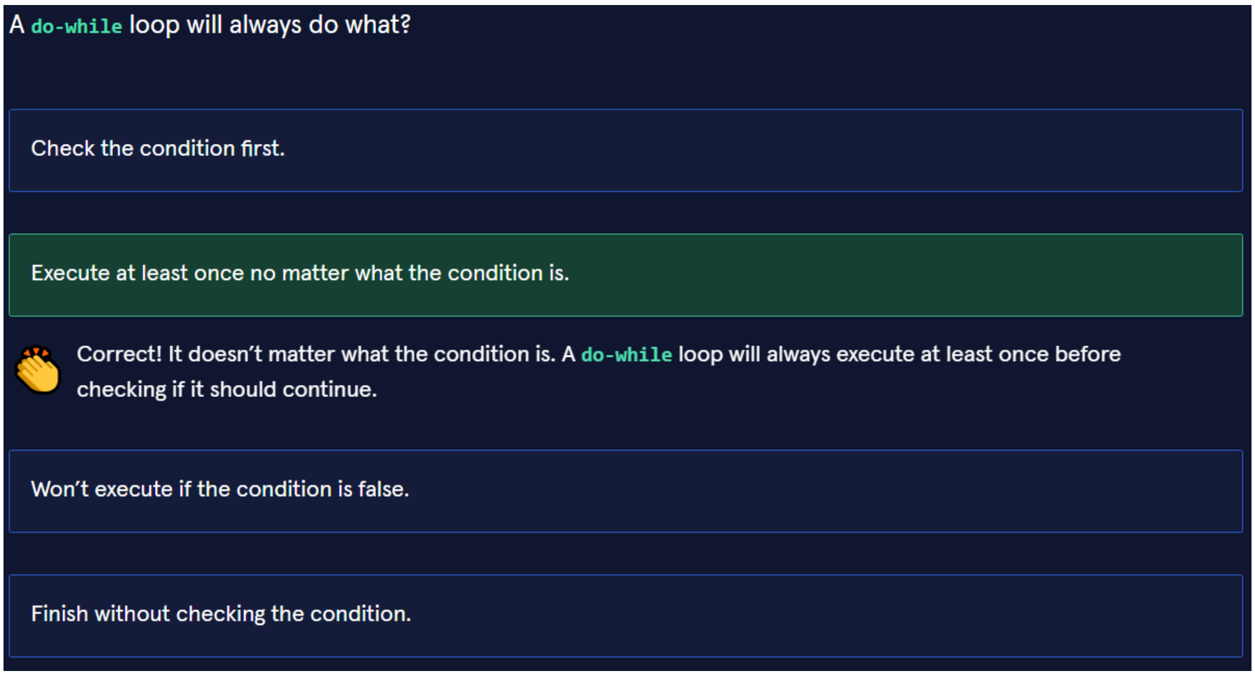


Which of the following is a run-time error?

- Forgetting a `;` somewhere.
- [✓ Division by zero.]

Correct! A division by zero is unexpected during the running of code, and can cause some very strange behavior.

- Using `Main()` instead of `main()`.
- Having an infinite loop.

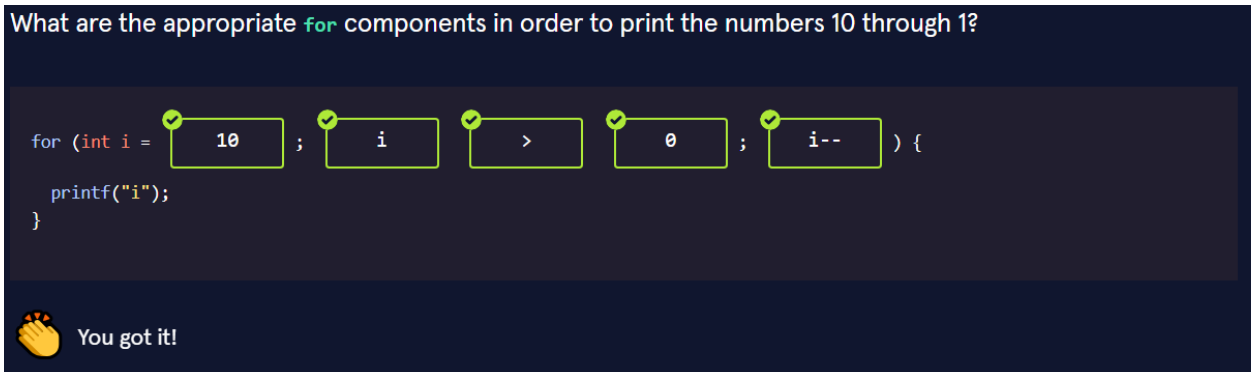


A `do-while` loop will always do what?

- Check the condition first.
- [✓ Execute at least once no matter what the condition is.]

Correct! It doesn't matter what the condition is. A `do-while` loop will always execute at least once before checking if it should continue.

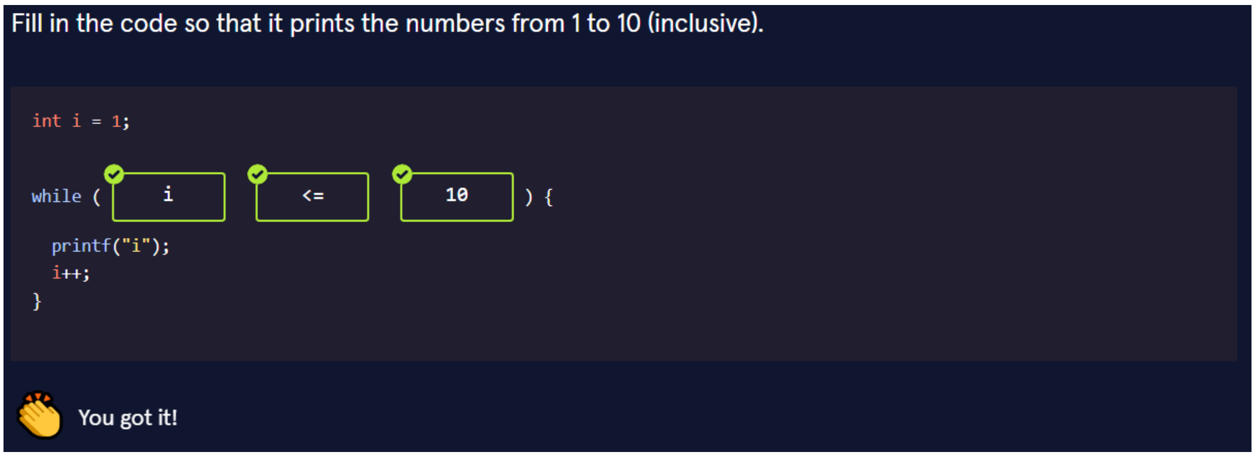
- Won't execute if the condition is false.
- Finish without checking the condition.



What are the appropriate `for` components in order to print the numbers 10 through 1?

```
for (int i = [10] ; [i] [>] [0] ; [i--] ) {
  printf("i");
}
```

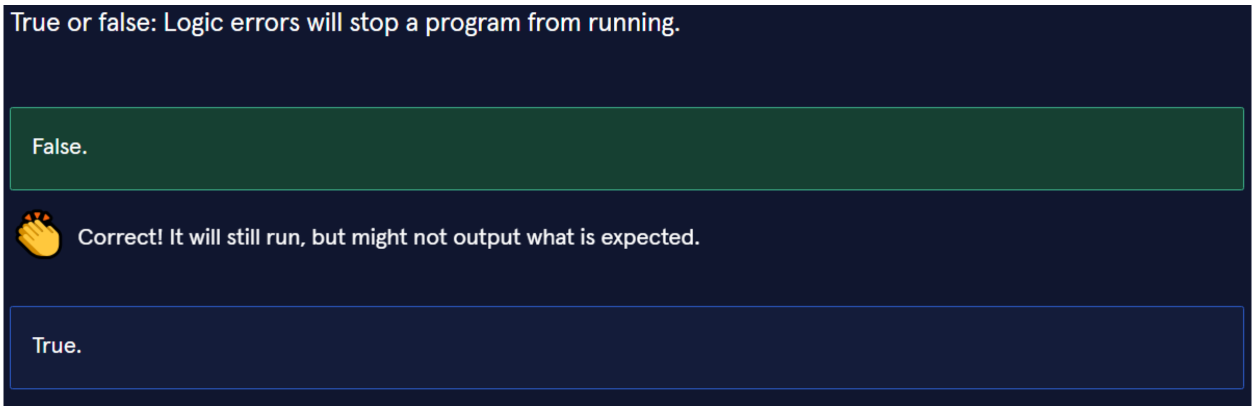
You got it!


Fill in the code so that it prints the numbers from 1 to 10 (inclusive).

```
int i = 1;

while ( i <= 10 ) {
  printf("i");
  i++;
}
```

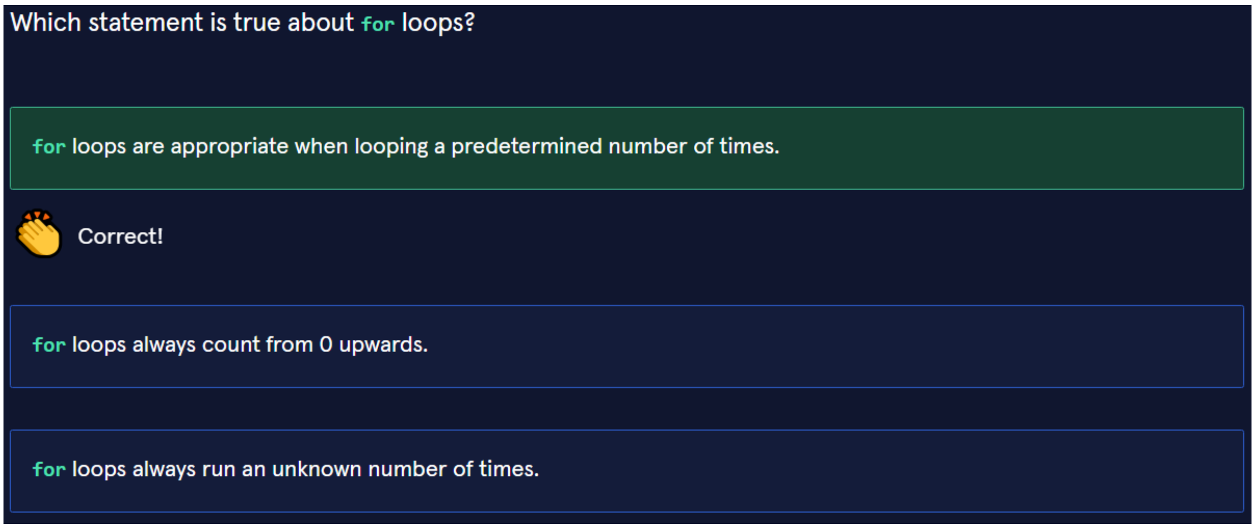
You got it!


True or false: Logic errors will stop a program from running.

False.

Correct! It will still run, but might not output what is expected.

True.

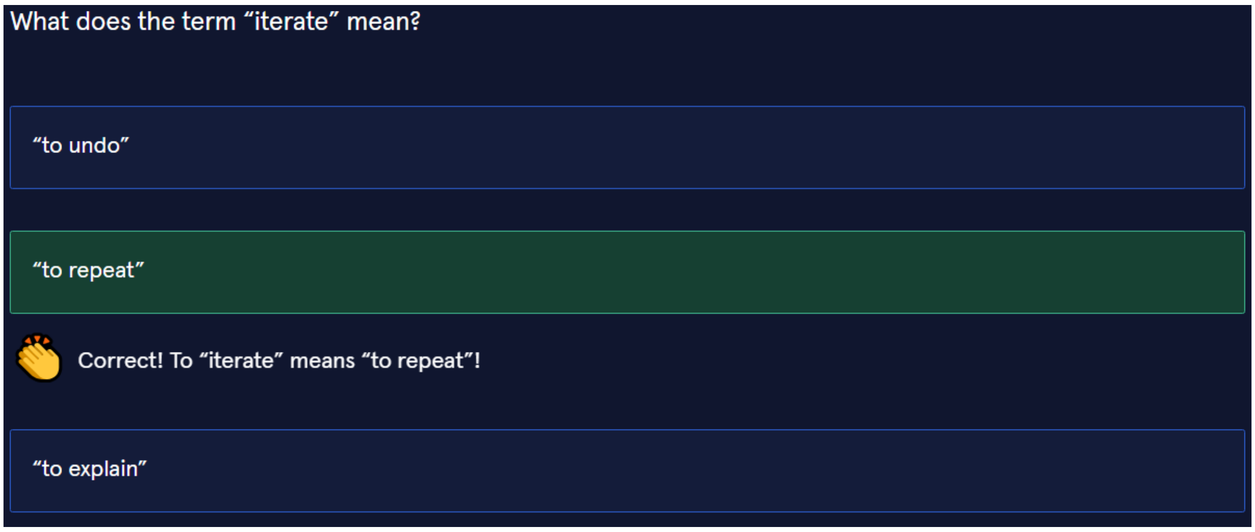

Which statement is true about `for` loops?

`for` loops are appropriate when looping a predetermined number of times.

Correct!

`for` loops always count from 0 upwards.

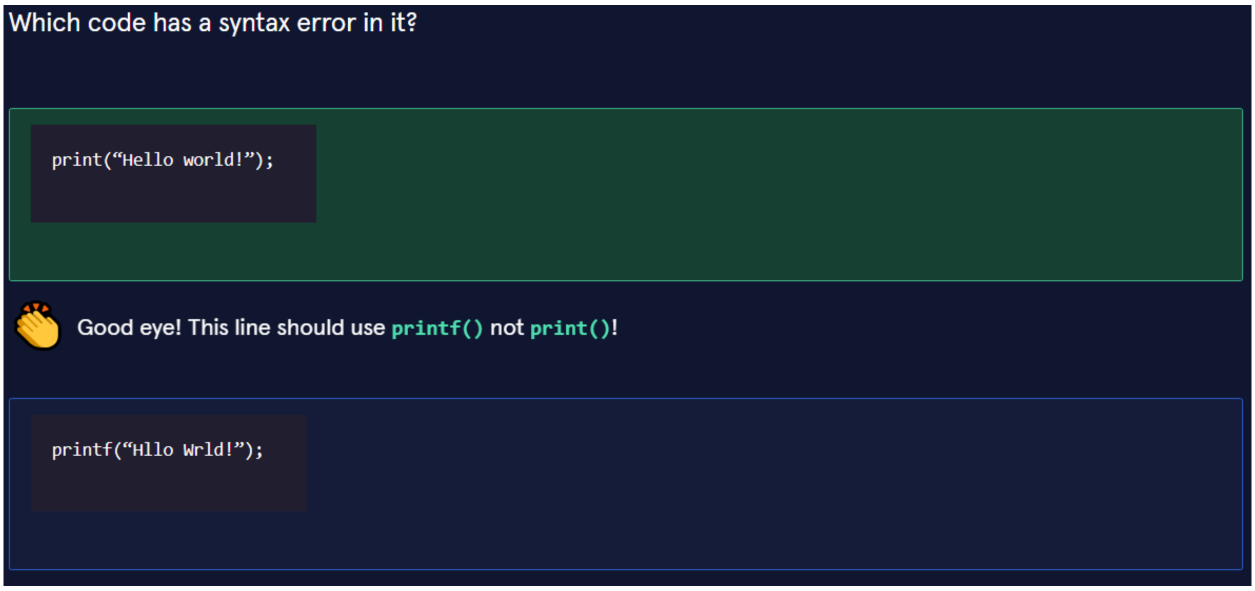
`for` loops always run an unknown number of times.


What does the term “iterate” mean?

- “to undo”
- [✓ “to repeat”]

👏 Correct! To “iterate” means “to repeat”!

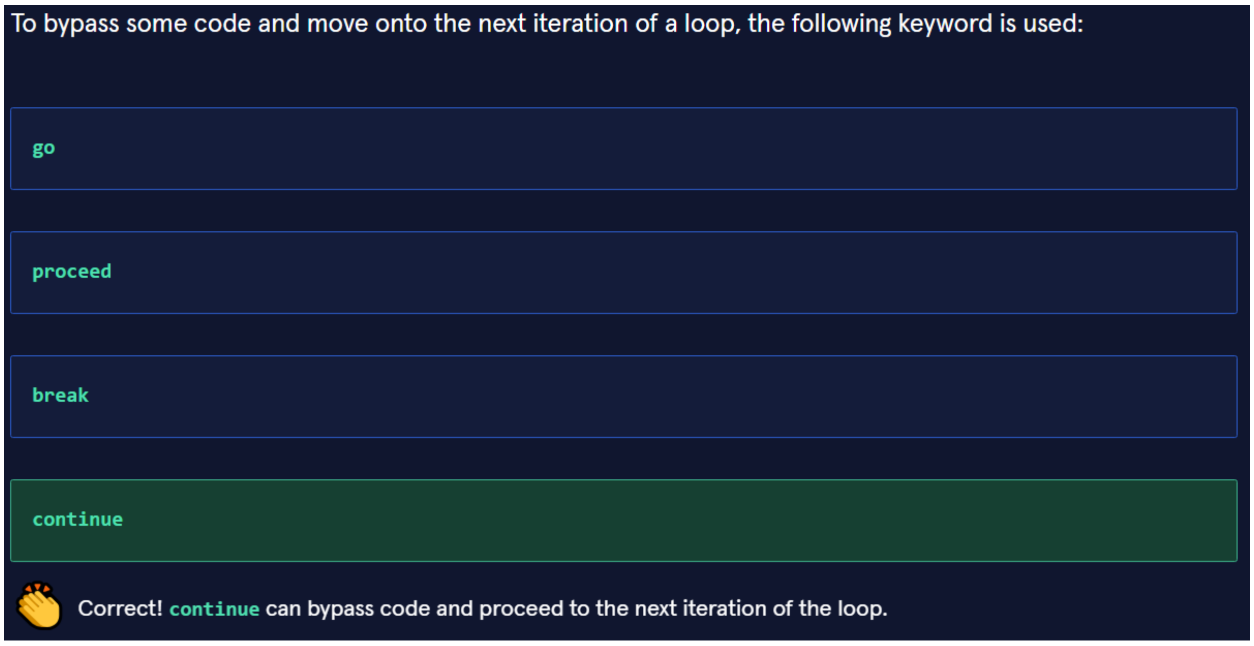
- “to explain”


Which code has a syntax error in it?

[✓
```
print(“Hello world!”);
```
]

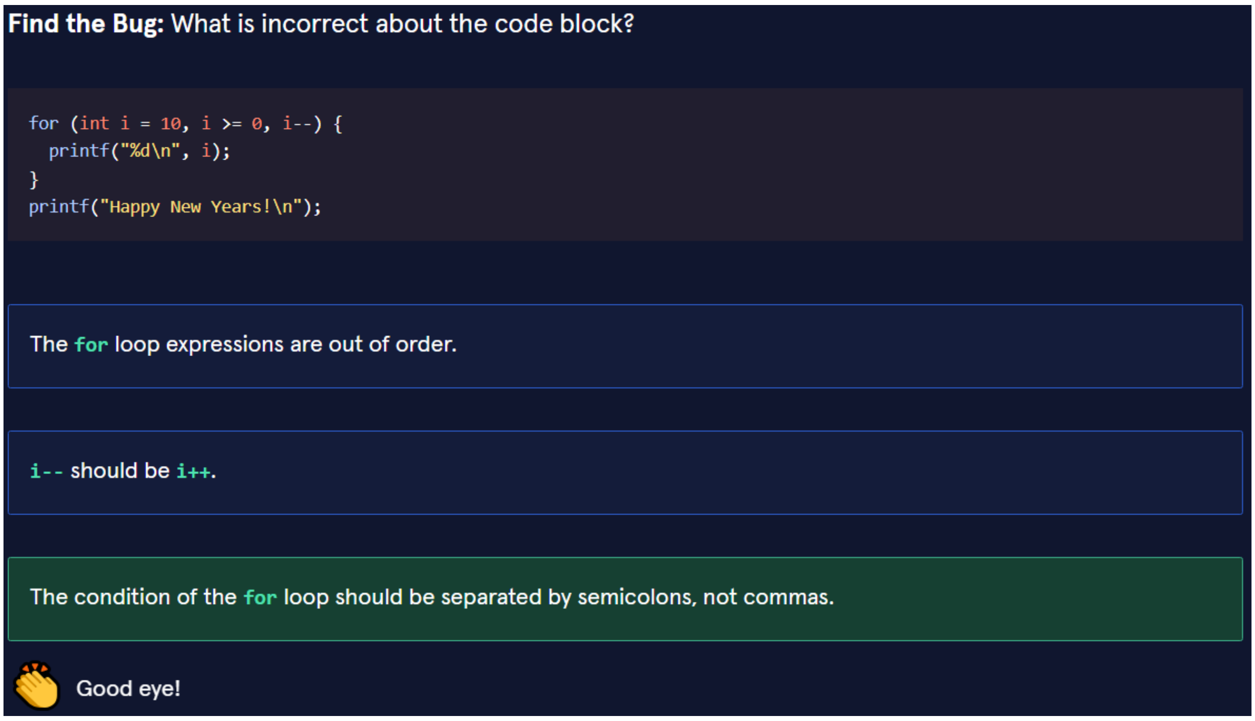
👏 Good eye! This line should use `printf()` not `print()`!

```
printf(“Hllo Wrld!”);
```


To bypass some code and move onto the next iteration of a loop, the following keyword is used:

- `go`
- `proceed`
- `break`
- [✓ `continue`]

👏 Correct! `continue` can bypass code and proceed to the next iteration of the loop.



Find the Bug: What is incorrect about the code block?

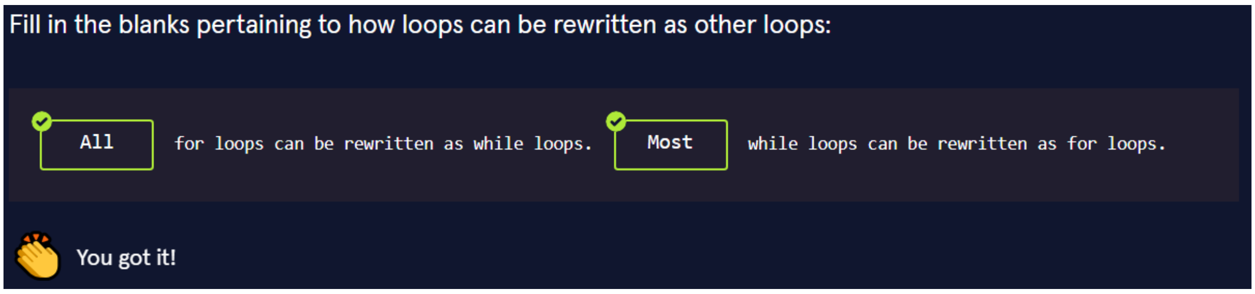
```
for (int i = 10, i >= 0, i--) {
  printf("%d\n", i);
}
printf("Happy New Years!\n");
```

The `for` loop expressions are out of order.

`i--` should be `i++`.

[✓ The condition of the `for` loop should be separated by semicolons, not commas.]

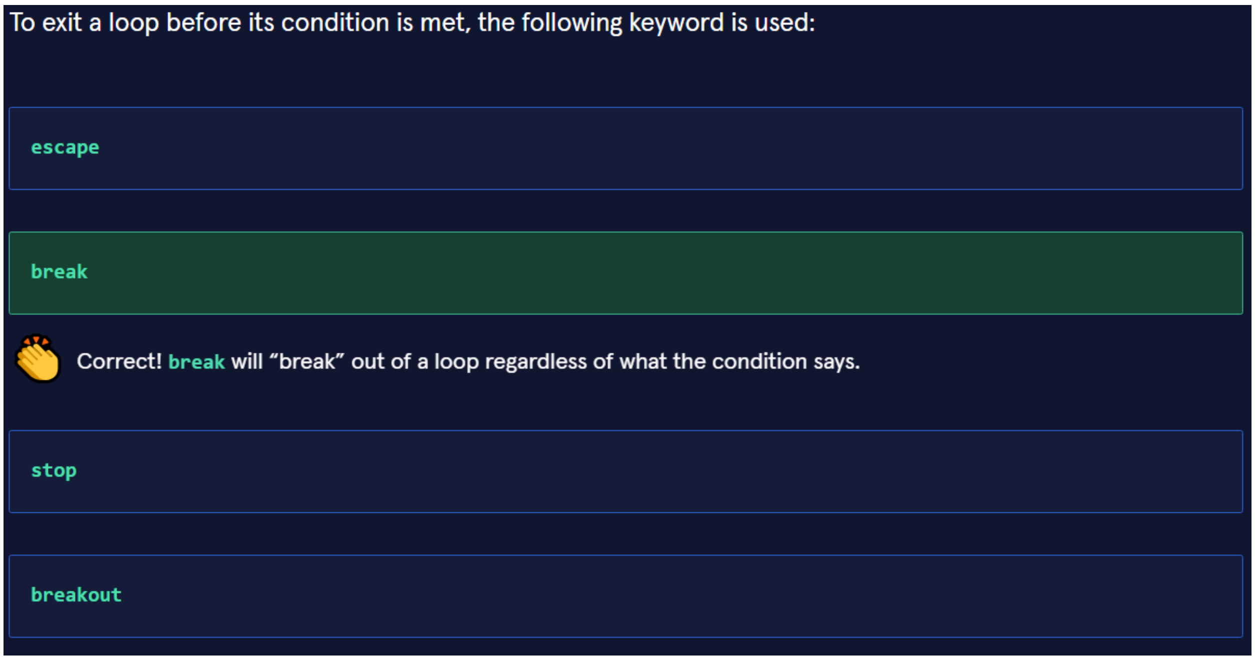
👏 Good eye!



Fill in the blanks pertaining to how loops can be rewritten as other loops:

`[✓ All] for loops can be rewritten as while loops. [✓ Most] while loops can be rewritten as for loops.`

👏 You got it!



To exit a loop before its condition is met, the following keyword is used:

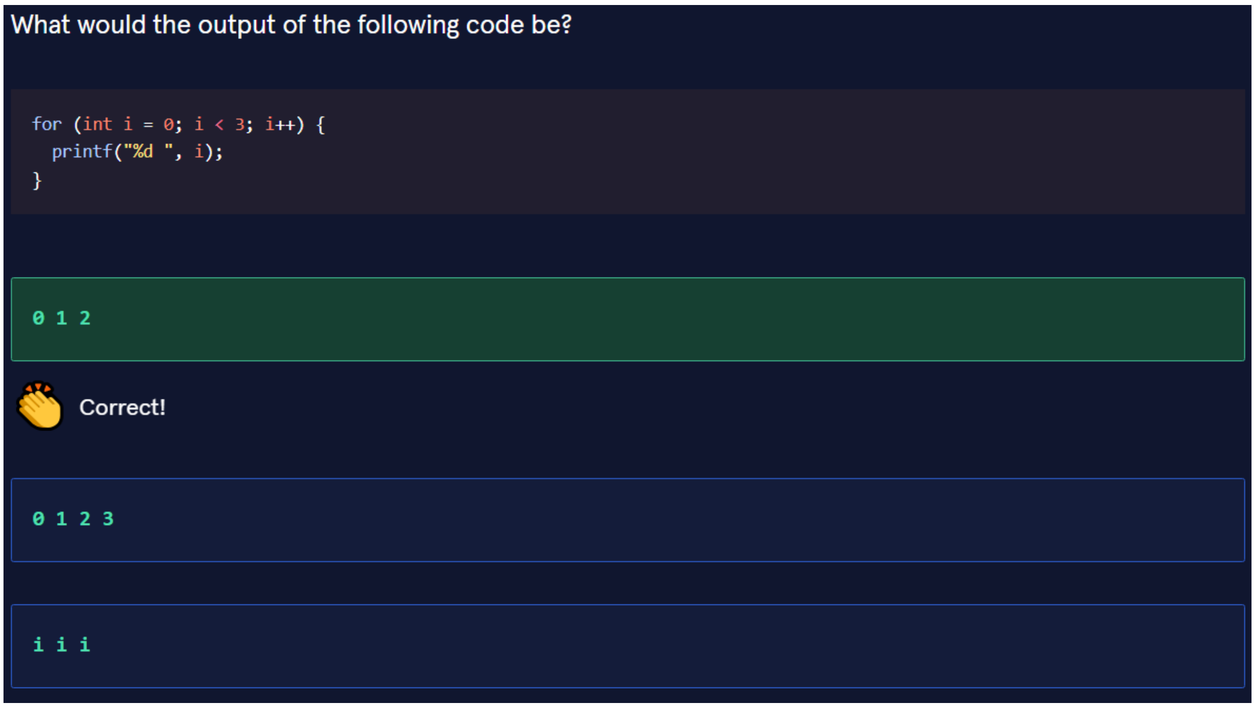
`escape`

[✓ `break`]

👏 Correct! `break` will “break” out of a loop regardless of what the condition says.

`stop`

`breakout`



What would the output of the following code be?

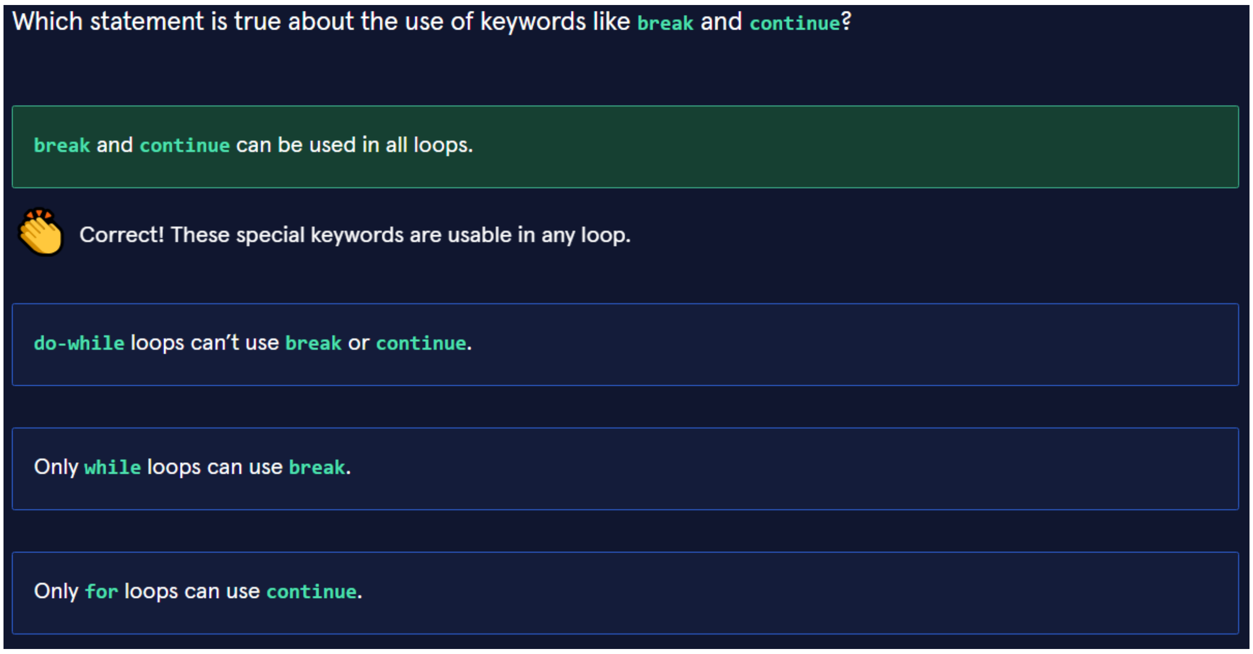
```
for (int i = 0; i < 3; i++) {
  printf("%d ", i);
}
```

[✓ `0 1 2`]

👏 Correct!

`0 1 2 3`

`i i i`



Which statement is true about the use of keywords like `break` and `continue`?

[✓ `break` and `continue` can be used in all loops.]

👏 Correct! These special keywords are usable in any loop.

`do-while` loops can’t use `break` or `continue`.

Only `while` loops can use `break`.

Only `for` loops can use `continue`.