QUIZ

Which of the following is true regarding an argument passed to a function as a pointer?

- Passing the argument to a function as a pointer can change the value it points to both inside and outside the function.

Correct! If the value it points to changes inside the function, this means the change is recorded wherever the variable can be accessed (including outside the function) because the change is made directly at th...
Show more

- Passing the argument to a function as a pointer can change the value it points to only within the function.
- None of the other three options are true.
- Passing the argument to a function as a pointer can change the value it points to only outside the function.

Complete the `abs()` function by providing the correct return type and using the correct logic inside the body.

```
int abs(int n) {
  if (n < 0) {
    return -n;
  }
  else return n;
}
```

You got it!

What is the output of the following code?

```
#include<stdio.h>

void f(int *p) {
  int b = 2;
  p = &b;
  printf("%d ", *p);
}

int main() {
  int a = 1;
  int *p = &a;
  f(p);
  printf("%d ", *p);
}
```

11

12

21

This is correct! The trick lies in knowing that a pointer being passed to a function is still being passed by value, and therefore the copy of the pointer inside the function now points to a different variabl...

Show more

This one's a bit tricky! Remember the `getRandomNumber(int a)` function you created earlier in the lesson? Instead of returning a random number from `1` to `a`, try adding a second parameter `b` to the function so that it now returns a random number between `a` and `b`:

```
int getRandomNumber(int a, int b ) {
  int randomNumber = rand() % ( b - a + 1) + a ;
  return randomNumber;
}
```

You got it!

Which of the following is true about passing an argument to a function by value?

Passing the argument to a function by value can change the original variable's value but not the value of the parameter variable.

Passing the argument to a function by value can change the value of the parameter variable but not the original variable's value.

Correct! As covered earlier in the lesson, when a function has received an argument by value, the function makes a copy of the argument's value and stores it in the parameter variable. We can safely alter the...
Show more

Which of the following is a function in C?

`time()`

Correct! As introduced earlier, when passed the argument `NULL`, the `time()` function will return the number of seconds passed since the date, January 1, 1970.

`math.h`

`print()`

`time.h`

Which of the following is NOT true about functions?

A function cannot take a different number of arguments each time it is called.

Indeed, this is not always true! Some functions are quite flexible with the number of arguments they can take. (Remember `printf()`?)

A function cannot return a type other than the one specified in the declaration.

A function can be called without any arguments.

Every function needs a return type.

Which of the following about a function prototype is NOT true?

It makes sure that the function is named accordingly in the function declaration.

It makes sure the function call is correct in terms of the number, order, and type of parameters.

It makes sure that the parameters are named accordingly in the function declaration.

This is indeed NOT true! The compiler is concerned with 3 things when it comes to function prototypes: 1) The function name 2) The return type 3) The parameter type(s). The names of the parameter are not nece... Show more

It makes sure the compiler is aware of the function signature before the body is implemented so that the function can be called without having to worry about whether the compiler is aware of the function definition.