

Variable Scope

Variables in C++ all have types. That is because you are telling the computer what kind of information the variable will hold, and how much memory space it should allocate to the variable. This is an important part of C++ that other higher-level programming languages (like Python) will not require from their programmer, since they deal with that for you in the background.

But there is one property of variables, that is (nearly) universal in programming languages: the variable's scope. That is to say, how transferrable a variable is from one part of a program to another. The scope of a variable essentially follows one rule: **The scope of a variable is restricted to the part of the program where the variable is declared.** And the advice for 'good practice' when thinking about where to declare any variable is also straightforward: **Try to give your variables as little scope as possible, or: avoid 'global' variables when you can.** The scope of a variable applies, regardless of the type of the variable.

Example Sketch 1: Explaining by Example

Below is an example sketch that will compile in the Arduino IDE. You can copy it over and try for yourself. Each of the variables is followed by a comment that explains its scope in the program.

```
int x = 2; //A 'global' variable that can be used anywhere
int y = 3; //A 'global' variable that can be used anywhere
void setup() { // the setup function runs once
    Serial.begin(9600);
    Serial.print(add(x,y));
    Serial.print(" is the sum");
    Serial.print(multiply(x,y));
    Serial.print(" is the product");
}
void loop() { // the setup function runs repeatedly
    long w = 5000000; //'w' can only be used in 'loop()'
    delay(w);
}
int add (int a, int b){ //'a' & 'b' can only be used in 'add()'
    int c = a + b; //this 'c' can only be used in 'add()'
    return c;
}
long multiply (int a, float b){ // same as above...
    long c = a * b; //this 'c' can only be used in 'multiply()'
    return c;
} // the 'c' in 'add()' and 'multiply()' are not in conflict
// because they have restricted scope that does not overlap
```

Explain the Serial Monitor behavior of the above sketch with written sentences:

Describe the output of this program on a serial monitor. (what text and time intervals?)

Answer here

There are several variables in the program example above. Use full sentences to describe the scope of each of the variables. Be sure to describe where they can and cannot be used in the whole program above. (Keep in mind the functions, and refer to them by name.) If the variable has more than one separated scope, explain that as well and use clear language.

Answer here

Name	Type	Use	Scope
x			
y			
w			
a			
b			
c			

What reason can you think of that would explain why an experienced programmer might give a new programmer this advice: "Always use the smallest scope you can for variables."

Answer here

How is this output achieved by the program? (Explain the logic of the program)

Answer Here

Example Sketch 2: Listen to Error Messages

Below is an example array implemented in a program. You can see that the array is implemented in the 'void loop()' so this code will run forever, while powered.

```
void setup() { // the setup function runs once
  int ledPin = 13;
  pinMode(ledPin, OUTPUT);
} //morse code for the letters "A.I."
int times[] = {250,80,750,80,250,80,250,2000};
Bool ledState;
void loop() { // will run forever, while there is power
  for(x = 0; x < 8; x++){
    digitalWrite(ledPin, ledState); // output on pin 13
    ledState = !ledState; //change ledState so it will flip
    delay(times[x]); //wait some time from the array
  }
}
```

Fill in the table of the variables in the program above. Add rows as needed to have a complete explanation of every one of the variables in the sketch shown here. *do this for as shown

Name	Type	Use	Scope
times[]	int	Array to hold delay times	Global variable

The example sketch above will receive a series of error messages when you go to compile it in the Arduino IDE. The error messages will be from the highlighted portions of the code (although it is harder to tell that from the IDE error messages themselves). In the space below, explain the nature of each highlighted error, and how it could be resolved by the programmer.

Answer here

Arduino's Own Explanations

[Scope Reference Page](#) << a written explanation of variable scope, and how it works on Arduino

[Variable Scope discussed on Arduino Forum](#) << The community knows...

[Some guy's video explanation on Youtube](#) << there are always a bunch of these, but...

Create your own Sketches

You don't know if you understand it, until you can create it from nothing...

Using what you have seen above, in the example sketches and from Arduino's reference materials. Create your own sketch that implements a while loop in some way. You have plenty of creative license in this goal, but you must make a sketch that successfully uses a while loop. A screenshot of your sketch and output is half of your response to these. A written explanation of the logic is also required to prove that you understand what you are doing with this work.

Primary tasks are required for all students. Secondary tasks are required to get a top grade.

- Primary tasks:

- successfully implement a global variable, used in more than one function

Screenshot & Explanation here

- successfully implement a local variable in the 'void loop()' function

Screenshot & Explanation here

- successfully implement a very local variable in a 'for loop'

Screenshot & Explanation here

- Moderate understanding secondary tasks:

- Show an error message for trying to use a variable out of scope

Screenshot & Explanation here

- Show an error message for trying to use an undeclared variable

Screenshot & Explanation here