

Chapter 6

Using Buzzer with Arduino

No, Theory today. Let's go to Arduino IDE.

Build this circuit and upload the below code: (Note: 1. Buzzer has a positive pin and a negative pin. Negative means GND.)

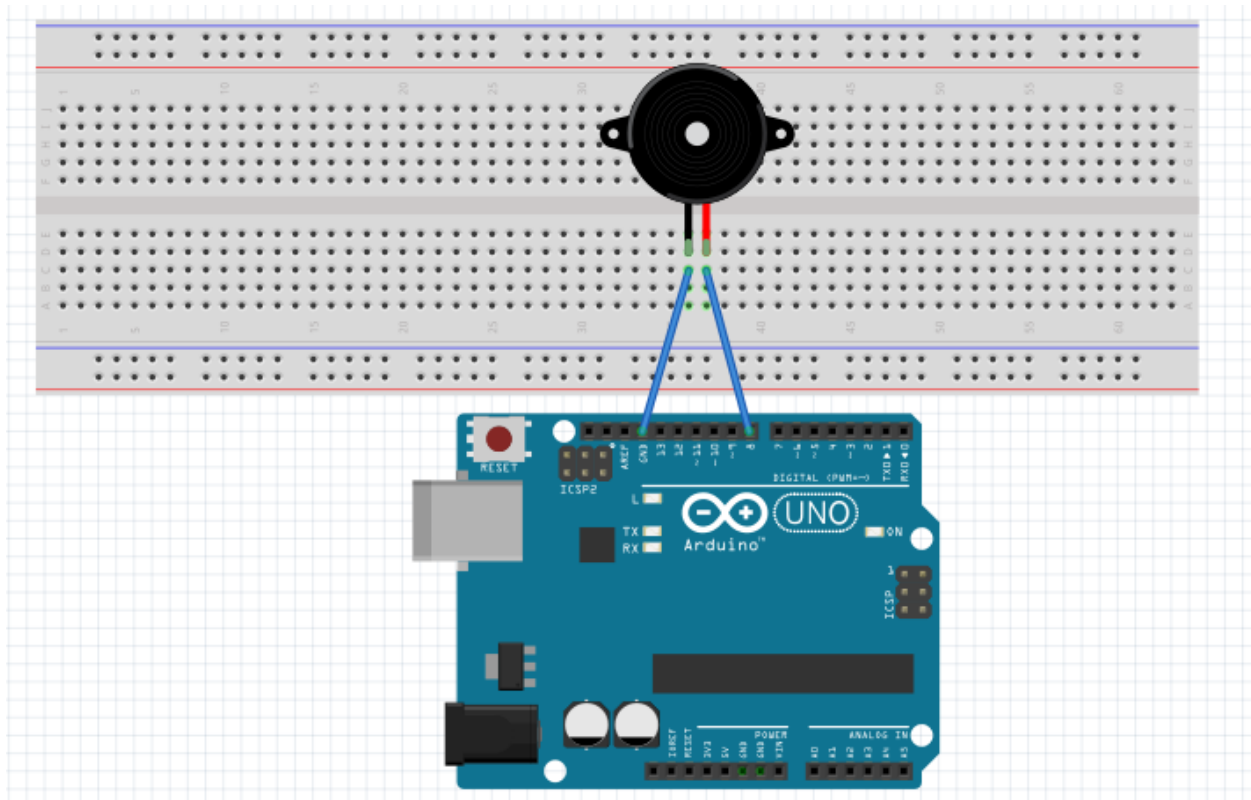


Fig 6.1: Buzzer connection with Arduino UNO (Your buzzer may not look like this)

A screenshot of the Arduino IDE interface. The top menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu is a toolbar with icons for checking, running, and uploading, followed by a dropdown menu showing 'Arduino Uno'. The main workspace displays a file named 'sketch_jun28a.ino' with the following code:

```
1 void setup() {  
2   pinMode(8, OUTPUT);  
3 }  
4  
5 void loop() {  
6   digitalWrite(8, HIGH);  
7   delay(500);  
8   digitalWrite(8, LOW);  
9   delay(500);  
10 }
```

After uploading you should hear some sound. Interesting, isn't it? If you face any problem, you know what to do.

Now, upload this code below keeping the circuit same:

A screenshot of the Arduino IDE interface, similar to the first one. The top menu bar and toolbar are the same. The dropdown menu still shows 'Arduino Uno'. The main workspace displays the same file 'sketch_jun28a.ino' but with a different code:

```
1 void setup() {  
2   pinMode(8, OUTPUT);  
3 }  
4  
5 void loop() {  
6   digitalWrite(8, HIGH);  
7   delay(100);  
8   digitalWrite(8, LOW);  
9   delay(200);  
10  digitalWrite(8, HIGH);  
11  delay(500);  
12  digitalWrite(8, LOW);  
13  delay(500);  
14 }
```

Do you hear anything? You have become a musician who can make music by changing code.

Now, make a music of your own by writing code.

Chapter 7

How to print something using Arduino

The first question is: Where shall we print some text? There are many ways. You can buy a LCD or OLED display as shown below.



Fig 6.1: 0.96 inch OLED display



Fig 6.2: 16x2 LCD display

But for now we are not going to buy anything. We will use our PC to show some text message. The message will be sent **by Arduino UNO through the Cable to the PC**. This communication between Arduino UNO and PC through cable is known as **Serial Communication**. (Don't memorize this as a definition of Serial Communication, this is not the exact one.) And in laptop where we will see the message sent by arduino UNO is known as Serial Monitor. Enough of theory, let's go to arduino UNO.

How to print something in Serial Monitor:

Upload this code to Arduino UNO. (That's println in code.) (Small of L)

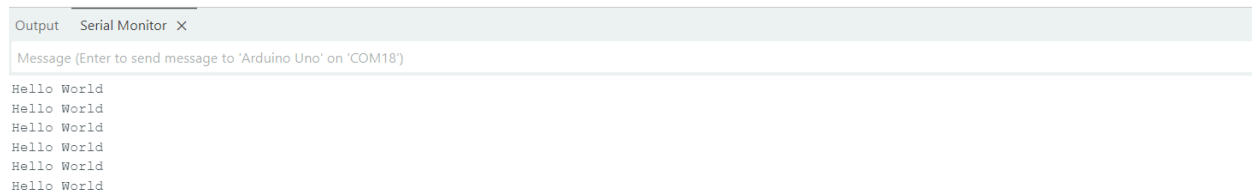


```
1 void setup() {
2   Serial.begin(9600);
3
4 }
5
6 void loop() {
7   Serial.println("Hello World");
8   delay(1000);
9
10 }
```

After uploading the open the Serial Monitor using the Icon Circular icon at the upper right of Arduino IDE. One that looks like microscope below:



If everything is correct you should see message being printed by Arduino UNO at the lower side of your screen. In this form:



```
Output Serial Monitor X
Message (Enter to send message to 'Arduino Uno' on 'COM18')
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
```

Hurrah! This is a very big milestone. There is a tradition that programmers start learning a language by printing Hello World at the first time. If you don't see the message or face any problem feel free to ask in our group.

OK. Now upload the below code to Arduino UNO: (Replace Your Name by your name.)

A screenshot of the Arduino IDE interface. The top menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu is a toolbar with icons for checking, running, and uploading, followed by a dropdown menu showing 'Arduino Uno'. The main workspace displays a file named 'sketch_jun28a.ino' with the following code:

```
1 void setup() {
2   Serial.begin(9600);
3
4 }
5
6 void loop() {
7   Serial.println("Your Name");
8   delay(1000);
9
10 }
```

See the output.

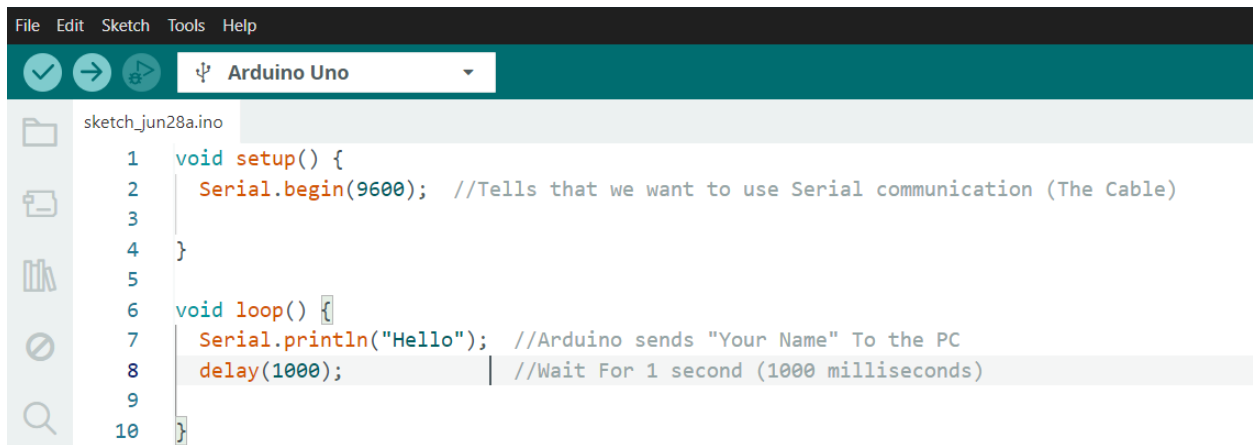
Explanation: (See the comments beside each line.)

A screenshot of the Arduino IDE interface, similar to the previous one, but with comments added to the code. The code is as follows:

```
1 void setup() {
2   Serial.begin(9600); //Tells that we want to use Serial communication (The Cable)
3
4 }
5
6 void loop() {
7   Serial.println("Your Name"); //Arduino sends "Your Name" To the PC
8   delay(1000); //Wait For 1 second (1000 milliseconds)
9
10 }
```

What is 9600? It is the BPS (bits per second) of serial communication. Note that in Serial monitor the BPS is also 9600. This BPS is also known as baud rate or transmission rate. (Don't memorize these terminologies, you will get familiar to them automatically.)

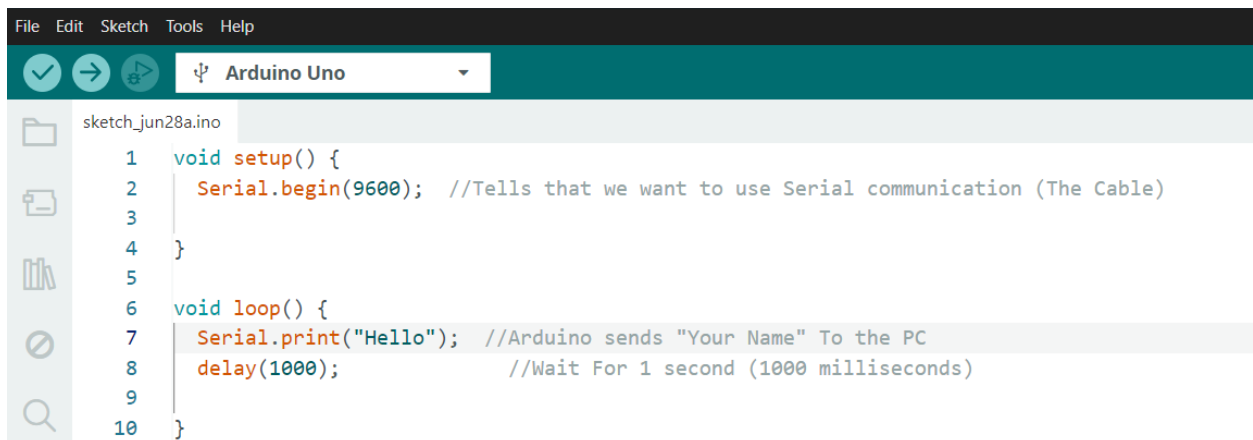
Now upload this code below:



```
File Edit Sketch Tools Help
sketch_jun28a.ino
1 void setup() {
2   Serial.begin(9600); //Tells that we want to use Serial communication (The Cable)
3
4 }
5
6 void loop() {
7   Serial.println("Hello"); //Arduino sends "Your Name" To the PC
8   delay(1000); //Wait For 1 second (1000 milliseconds)
9
10 }
```

See the output. Note that every time Hello is being printed in a new line.

Now upload this code below: (print only not println).



```
File Edit Sketch Tools Help
sketch_jun28a.ino
1 void setup() {
2   Serial.begin(9600); //Tells that we want to use Serial communication (The Cable)
3
4 }
5
6 void loop() {
7   Serial.print("Hello"); //Arduino sends "Your Name" To the PC
8   delay(1000); //Wait For 1 second (1000 milliseconds)
9
10 }
```

See the output. Do you see any difference?

Now, upload this code below: (We are making things complex slowly)

```
1 void setup() {
2   Serial.begin(9600);
3 }
4
5 void loop() {
6   Serial.print("Hello");
7   Serial.print(" "); //This line prints a space
8   Serial.println("My name is Your_name");
9   delay(1000);
10 }
```

See the output. Why the output is like this? (Hints: difference between println and print). (**ln stands for new line**).

Task 01:

Try printing this message on your own: (replace name by your name)

Hello World! I am name. I am learning arduino.

Chapter 8

Introduction to C++ programming in Arduino

Don't be afraid we are not going to learn all those complicated things. Only these below topics will be enough for us as we are just beginning.

1. Different variables: int, char
2. User defined function
3. Basic math: Plus, Minus, Multiplication, Division, Reminder.
4. Condition: if...else
5. Loop: while, for

Variables in Arduino

Integer type variables: These types of variables can hold integer numbers.

Upload the below code to arduino:



```
File Edit Sketch Tools Help
[Checkmark] [Next] [Upload] [Serial Monitor] Arduino Uno
sketch_jun28a.ino
1  int val = 500;
2
3  void setup() {
4      Serial.begin(9600);
5  }
6
7  void loop() {
8      Serial.println(val);
9      delay(1000);
10 }
```

What do you see in the serial monitor?

Now upload this code below. See the output in serial monitor and try to understand on your own.



```
File Edit Sketch Tools Help
✓ → ⚙️ Arduino Uno
sketch_jun28a.ino
1 int value1 = 500;
2 int value2 = 1000;
3
4 void setup() {
5   Serial.begin(9600);
6 }
7
8 void loop() {
9   Serial.print("First number is: ");
10  Serial.println(value1);
11  Serial.print("Second number is: ");
12  Serial.println(value2);
13  delay(1000);
14 }
```

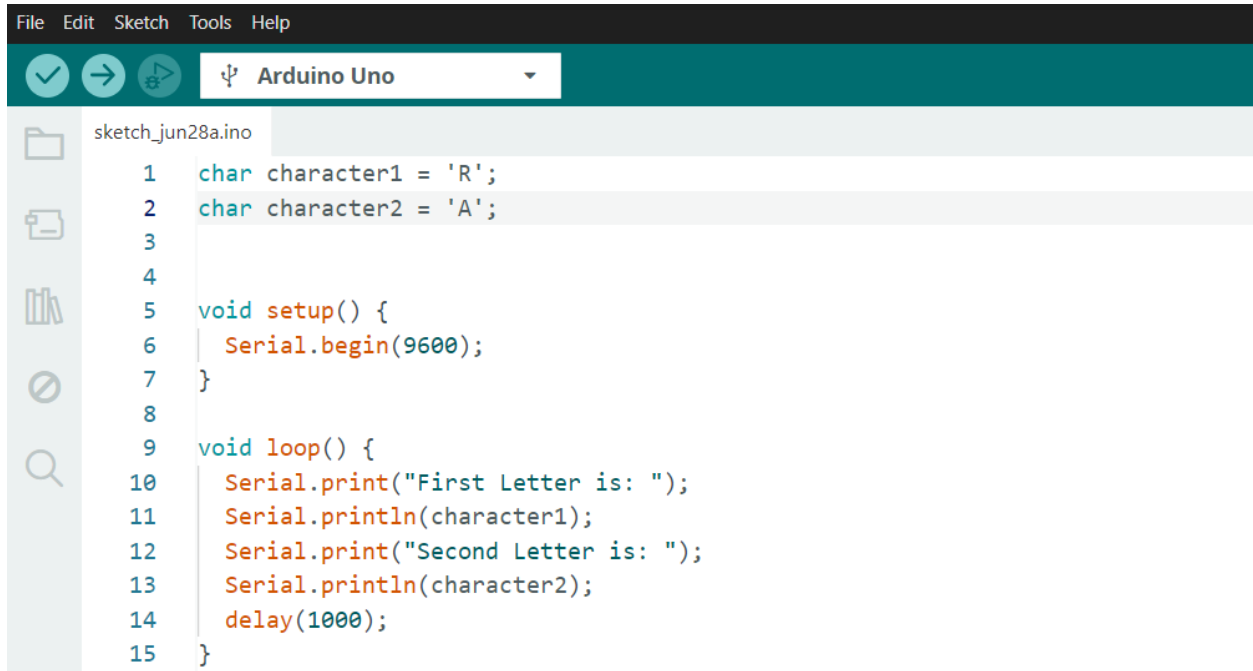
Character type variables: These type of variables can hold a letter inside them.

Upload this code below and see the output:



```
File Edit Sketch Tools Help
✓ → ⚙️ Arduino Uno
sketch_jun28a.ino
1 char val = 'R'; // Note R is in comma. This is the rule. Don't ask why now. You will realise soon.
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8   Serial.println(val);
9   delay(1000);
10 }
```

Now upload this code below and see the output: (If it does not make sense, don't panic).



```
File Edit Sketch Tools Help
ψ Arduino Uno
sketch_jun28a.ino
1 char character1 = 'R';
2 char character2 = 'A';
3
4
5 void setup() {
6   Serial.begin(9600);
7 }
8
9 void loop() {
10  Serial.print("First Letter is: ");
11  Serial.println(character1);
12  Serial.print("Second Letter is: ");
13  Serial.println(character2);
14  delay(1000);
15 }
```

See the output. And try to understand how to print a variable and how to print a text. The first line (at line 10) is a text and it is inside a “ “ double comma. But the character1 (at line 11) is a variable and it is not inside a “ “ double quotation.

Basic Math in Arduino:

Summation (Plus operation): Upload this code to arduino UNO and see the output in the serial monitor.

A screenshot of the Arduino IDE interface. The top menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu is a toolbar with icons for checking, running, and uploading, followed by a dropdown menu showing 'Arduino Uno'. The main workspace displays a sketch named 'sketch_jun28a.ino' with the following code:

```
1  int num1 = 50;
2  int num2 = 100;
3  int summation;
4  void setup() {
5      Serial.begin(9600);
6  }
7
8  void loop() {
9      summation = num1 + num2;
10     Serial.print("The answer is: ");
11     Serial.println(summation);
12     delay(1000);
13 }
```

Try to add 300 and 350 on your own and print the answer in the serial monitor.

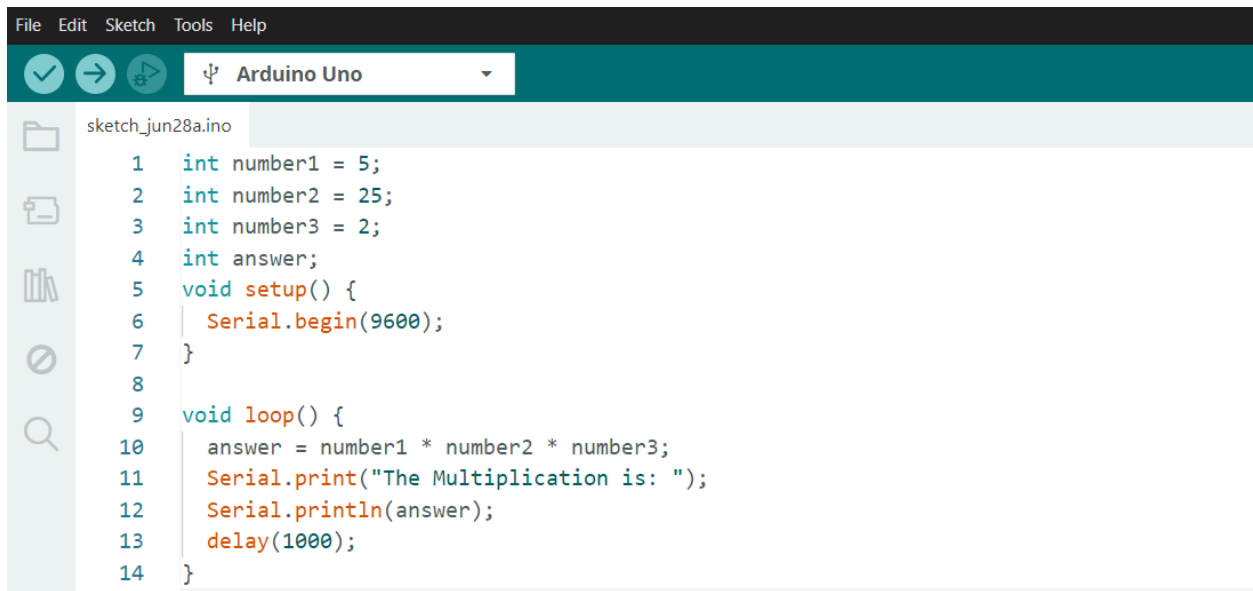
Subtraction: Upload this code to arduino and see the output.

A screenshot of the Arduino IDE interface, similar to the first one. The top menu bar and toolbar are the same. The main workspace displays a sketch named 'sketch_jun28a.ino' with the following code:

```
1  int num1 = 500;
2  int num2 = 190;
3  int answer;
4  void setup() {
5      Serial.begin(9600);
6  }
7
8  void loop() {
9      answer = num1 - num2;
10     Serial.print("The subtraction is: ");
11     Serial.println(answer);
12     delay(1000);
13 }
```

So, the moral of the story is arduino can do some math. You will be ashtonished to hear that arduino can do this types of plus, minus operations thousands of times in a second though it has only 2KB of RAM.

Multiplication: Upload this code to arduino and see the output.

A screenshot of the Arduino IDE interface. The top menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu is a toolbar with icons for checking, running, and uploading code, along with a dropdown menu currently set to 'Arduino Uno'. The main workspace shows a file named 'sketch_jun28a.ino' with the following C++ code:

```
1 int number1 = 5;
2 int number2 = 25;
3 int number3 = 2;
4 int answer;
5 void setup() {
6   Serial.begin(9600);
7 }
8
9 void loop() {
10   answer = number1 * number2 * number3;
11   Serial.print("The Multiplication is: ");
12   Serial.println(answer);
13   delay(1000);
14 }
```

So, Arduio can do multiplication. You can multiply 2 number, 3 number or any amount of numbers at a single time.

Division: Upload this code to arduino and see the output.



```
File Edit Sketch Tools Help
sketch_jun28a.ino
1 int number1 = 600;
2 int number2 = 40;
3
4 int answer1;
5 void setup() {
6   Serial.begin(9600);
7 }
8
9 void loop() {
10  answer1 = number1 / number2;
11  Serial.print("The division is: ");
12  Serial.println(answer1);
13  delay(1000);
14 }
```

Let me show you a problem:

Upload this code below. Do the division on your own and match the results.



```
File Edit Sketch Tools Help
sketch_jun28a.ino
1 int number1 = 50;
2 int number2 = 4;
3 int answer1;
4
5 void setup() {
6   Serial.begin(9600);
7 }
8
9 void loop() {
10  answer1 = number1 / number2;
11  Serial.print("The division is: ");
12  Serial.println(answer1);
13  delay(1000);
14 }
```

The answer should be 12.5 but arduino is telling that the answer is 12. That is a problem of using int type variable alway. This int type variables can not take a number with a point like 12.5. Try to find a soluton. If you can not find the solution, let it go. 12 and 12.5 is almost same to an engineer.

Reminder: (Banglai vagshesh) Upload this code below and see the output.

Enough programming. Let's solve some questions.

Assignment 02:

1. Solve these below maths using arduino: (if you know already, you can skip.)
 - a. $100 + 700 = ?$ (I know you can do it by just using your brain's calculator, but our target is to make the arduino work for us.)
 - b. $3000 - 2100 = ?$
 - c. $300 * 6 = ?$
 - d. $110 / 10 = ?$
 - e. $100 + 120 - 200 = ?$
 - f. $100 - 200 + 300 * 2 = ?$ (Remember this math from class 2 ?)
 - g. $300*2 + 200*3 + 400/2 = ?$ (Seems hard, isn't it?)
 - h. $(20 + 100) \% 7 = ?$
 - i. $(200*40) / (100*20) = ?$
 - j. $((300*5) \% (200/4)) + (30 - 20)*(30+20) = ?$

Conditions:

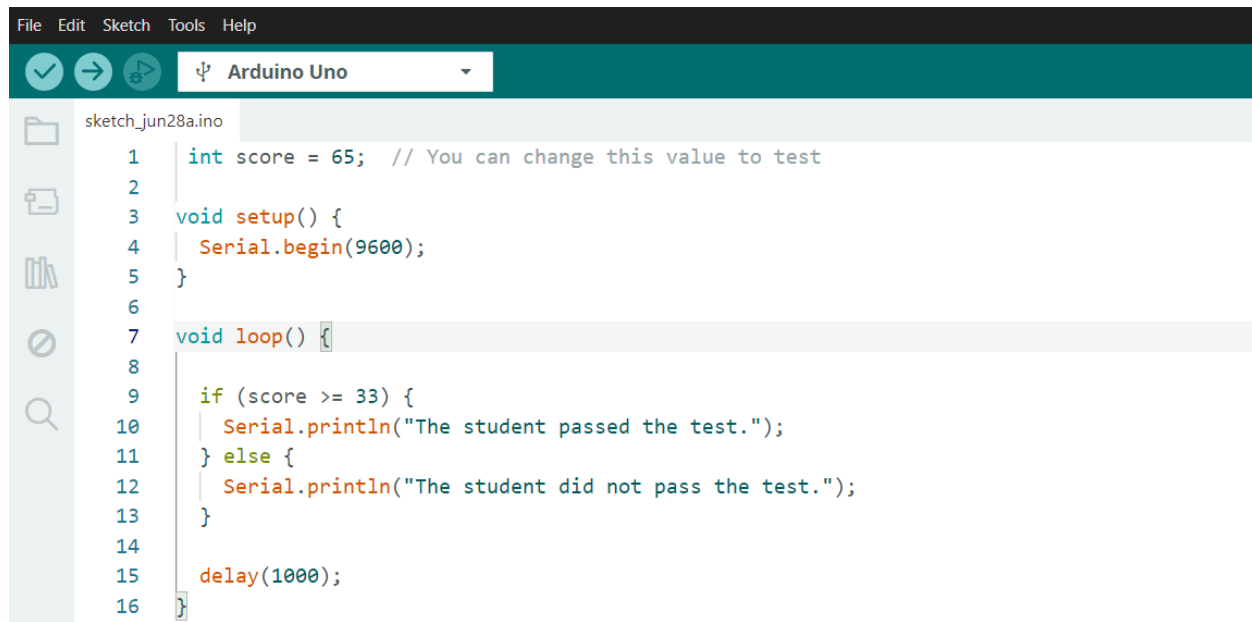
Upload this code below and see the output:



```
File Edit Sketch Tools Help
sketch_jun28a.ino
1  int temperature = 28; // You can change this value to test
2
3  void setup() {
4      Serial.begin(9600);
5  }
6
7  void loop() {
8
9      if (temperature > 30) {
10         Serial.println("The weather is quite warm.");
11     } else {
12         Serial.println("The weather feels pleasant.");
13     }
14
15     delay(1000);
16 }
```

Now change the temperature to 35 and upload the code again and see the result.

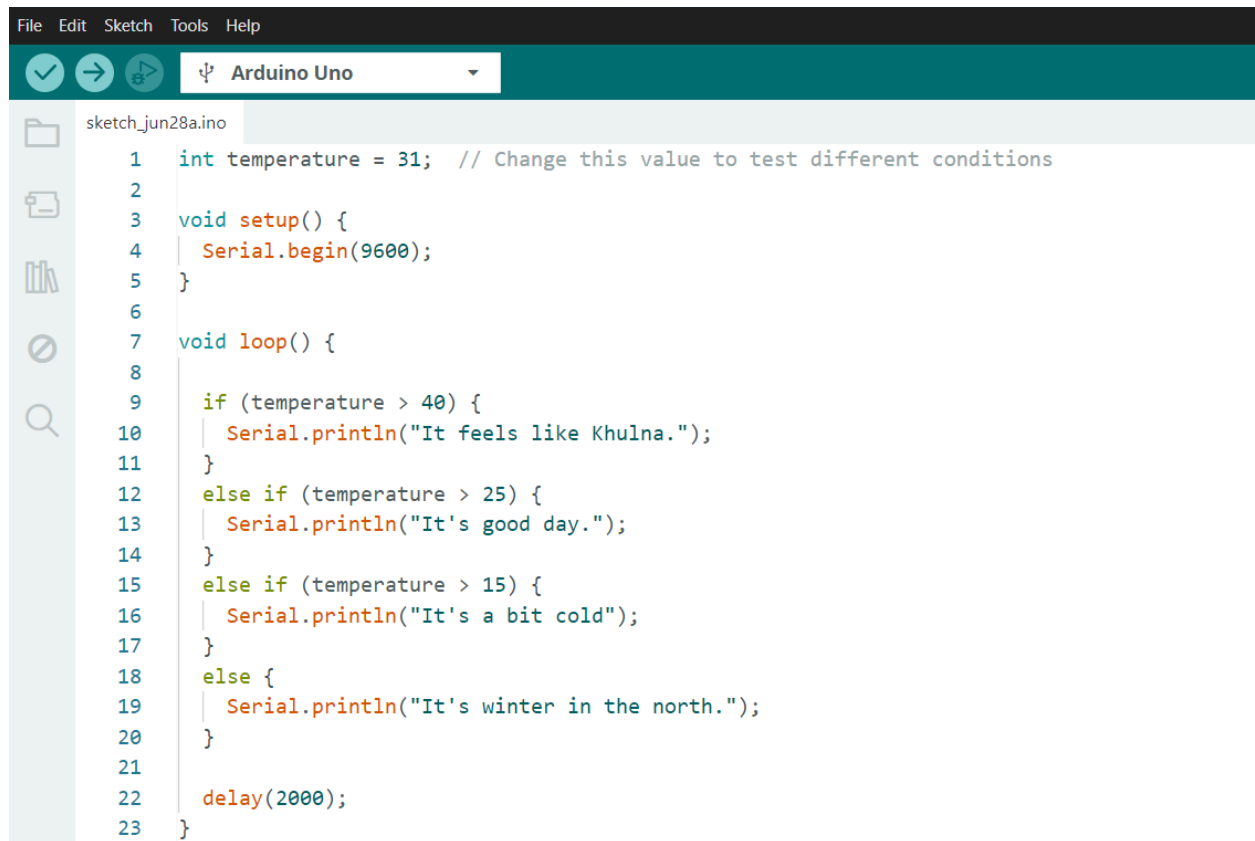
Now upload this code and see what happens:



```
File Edit Sketch Tools Help
sketch_jun28a.ino
1  int score = 65; // You can change this value to test
2
3  void setup() {
4      Serial.begin(9600);
5  }
6
7  void loop() {
8
9      if (score >= 33) {
10         Serial.println("The student passed the test.");
11     } else {
12         Serial.println("The student did not pass the test.");
13     }
14
15     delay(1000);
16 }
```

Hope you have a little idea about if...else.

Now try this code below and learn if...else if...else:

A screenshot of the Arduino IDE interface. The top menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu bar is a toolbar with icons for checking, running, and uploading code, followed by a dropdown menu showing 'Arduino Uno'. The main workspace displays a sketch named 'sketch_jun28a.ino'. The code is as follows:

```
1 int temperature = 31; // Change this value to test different conditions
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8
9   if (temperature > 40) {
10    Serial.println("It feels like Khulna.");
11  }
12  else if (temperature > 25) {
13    Serial.println("It's good day.");
14  }
15  else if (temperature > 15) {
16    Serial.println("It's a bit cold");
17  }
18  else {
19    Serial.println("It's winter in the north.");
20  }
21
22  delay(2000);
23 }
```

Now try this code below:

```
1 int score = 88; // You can change this value to test
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8   if (score >= 90 ) {
9     Serial.println("Grade: A+");
10  } else if (score >= 85) {
11    Serial.println("Grade: A");
12  } else if (score >= 80) {
13    Serial.println("Grade: A-");
14  } else if (score >= 75) {
15    Serial.println("Grade: B+");
16  } else if (score >= 70) {
17    Serial.println("Grade: B");
18  } else if (score >= 65) {
19    Serial.println("Grade: B-");
20  } else if (score >= 60) {
21    Serial.println("Grade: C");
22  } else if (score >= 50) {
23    Serial.println("Grade: D");
24  } else {
25    Serial.println("Grade: F");
26  }
27  delay(2000); // Check every 2 seconds
28 }
```

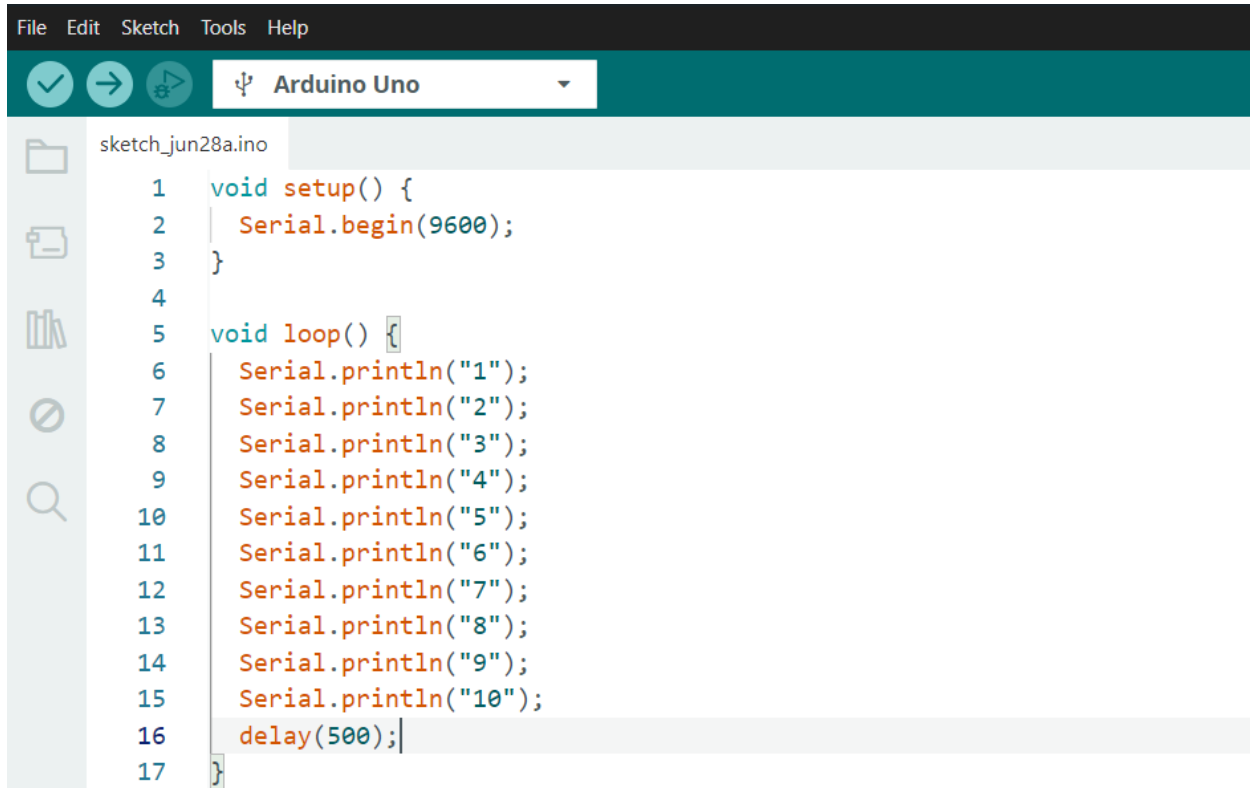
Sign of these condition:

1. == means if these two are equal (two = sign).
2. >= means greater than or equal.
3. <= means less than or equal.
4. < means less than.
5. > means greater than.

Also there are some operations like and, or, not that are quite useful. But we will learn them later.

Loop in ardino:

Suppose we want to print from 1 to 10 in the serial monitor. Here is the code:



```
File Edit Sketch Tools Help
ψ Arduino Uno
sketch_jun28a.ino
1 void setup() {
2   Serial.begin(9600);
3 }
4
5 void loop() {
6   Serial.println("1");
7   Serial.println("2");
8   Serial.println("3");
9   Serial.println("4");
10  Serial.println("5");
11  Serial.println("6");
12  Serial.println("7");
13  Serial.println("8");
14  Serial.println("9");
15  Serial.println("10");
16  delay(500);
17 }
```

Now what will you do if you want to print from 1 to 100? You have to write that `Serial.println()` 100 times!!!! But good news is that there is a better solution. The solution is running loop for 10 times or 100 times.

While loop:

Write and upload this code to arduino UNO.

The image shows the Arduino IDE interface. At the top, there's a menu bar with 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below it is a toolbar with icons for checking, running, and uploading, along with a dropdown menu showing 'Arduino Uno'. The main workspace displays a file named 'sketch_jun28a.ino'. The code in the editor is:

```
1 int number = 1;
2
3 void setup() {
4   Serial.begin(9600);
5
6   while (number <= 10) {
7     Serial.println(number);
8     number = number + 1;
9   }
10 }
11
12 void loop() {
13
14 }
```

Why the while is in void setup() {} . Lets put this in void loop() {} and try to understand what is happening:

Upload this code:

The image shows the same Arduino IDE interface as before. The code in 'sketch_jun28a.ino' has been modified to:

```
1 int number = 1;
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8
9   while (number <= 10) {
10     Serial.println(number);
11     number = number + 1;
12   }
13   delay(2000);
14
15 }
```

We can do this same thing using **for** loop. Let's try that:

Upload this code below:

The image shows the Arduino IDE interface. At the top is a menu bar with 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below it is a toolbar with icons for checking, running, and uploading, followed by a dropdown menu set to 'Arduino Uno'. The main workspace shows a file named 'sketch_jun28a.ino' with the following code:

```
1 void setup() {
2   Serial.begin(9600);
3
4   for (int number = 1; number <= 10; number = number + 1) {
5     Serial.println(number);
6   }
7 }
8
9 void loop() {
10 }
```

Again why the **for (; ;)** is in `void setup(){} ?` Let's put this in `void loop(){}.`

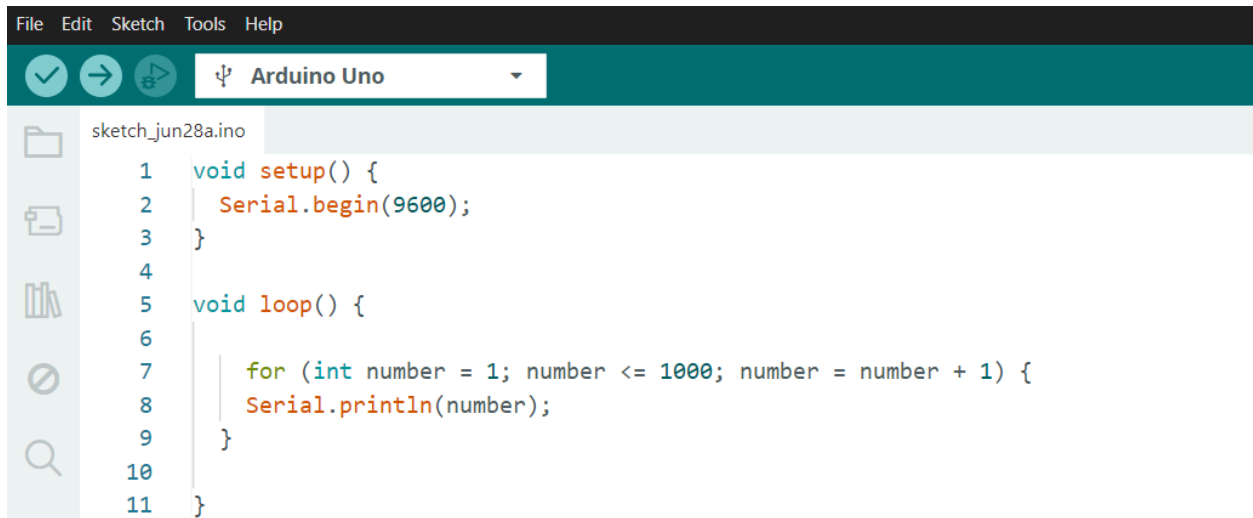
Upload this code below and see the output:

The image shows the Arduino IDE interface. At the top is a menu bar with 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below it is a toolbar with icons for checking, running, and uploading, followed by a dropdown menu set to 'Arduino Uno'. The main workspace shows a file named 'sketch_jun28a.ino' with the following code:

```
1 void setup() {
2   Serial.begin(9600);
3 }
4
5 void loop() {
6
7   for (int number = 1; number <= 10; number = number + 1) {
8     Serial.println(number);
9   }
10
11 }
```

Seems boring right? The core theory of something is most of the time a boring thing. But things become interesting when you start to apply them in a real world.

Why printing from 1 to 10. Let's print from 1 to 1000. This is the code :



```
1 void setup() {  
2   Serial.begin(9600);  
3 }  
4  
5 void loop() {  
6  
7   for (int number = 1; number <= 1000; number = number + 1) {  
8     Serial.println(number);  
9   }  
10  
11 }
```

As in the upper code, there is no delay so the printing will happen at a very fast rate. Even it may be faster than our eye's capability.

User Defined Function in Arduino:

So, You write code in arduino IDE. And only you use them. But this is not going to be true. Sometimes you will give others your code. And they will understand nothing because they don't know what your brain was thinking when you were writing the code.

Using **user defined function** is the way to help other people understand your code.

Let's go to our first code. LED blinking (on-off).

Upload this code below: (You have done this previously)

A screenshot of the Arduino IDE interface. The top menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu is a toolbar with icons for a checkmark, a right arrow, a play button, and a USB symbol. A dropdown menu shows 'Arduino Uno'. The left sidebar contains icons for a folder, a document, a book, a circle with a slash, and a magnifying glass. The main editor window shows a file named 'sketch_jun28a.ino' with the following code:

```
1 void setup() {  
2   pinMode(13, OUTPUT);  
3 }  
4  
5 void loop() {  
6   digitalWrite(13, HIGH);  
7   delay(1000);  
8   digitalWrite(13, LOW);  
9   delay(1000);  
10 }
```

You see this code is quite hard to understand. Now let's use a user defined function to write the same code:

Upload this code and look at the beauty of this code now:



```
1 void setup() {
2   pinMode(13, OUTPUT);
3 }
4
5 void loop() {
6   LED_on_off();
7 }
8
9
10
11 void LED_on_off(){
12   digitalWrite(13, HIGH);
13   delay(1000);
14   digitalWrite(13, LOW);
15   delay(1000);
16 }
```

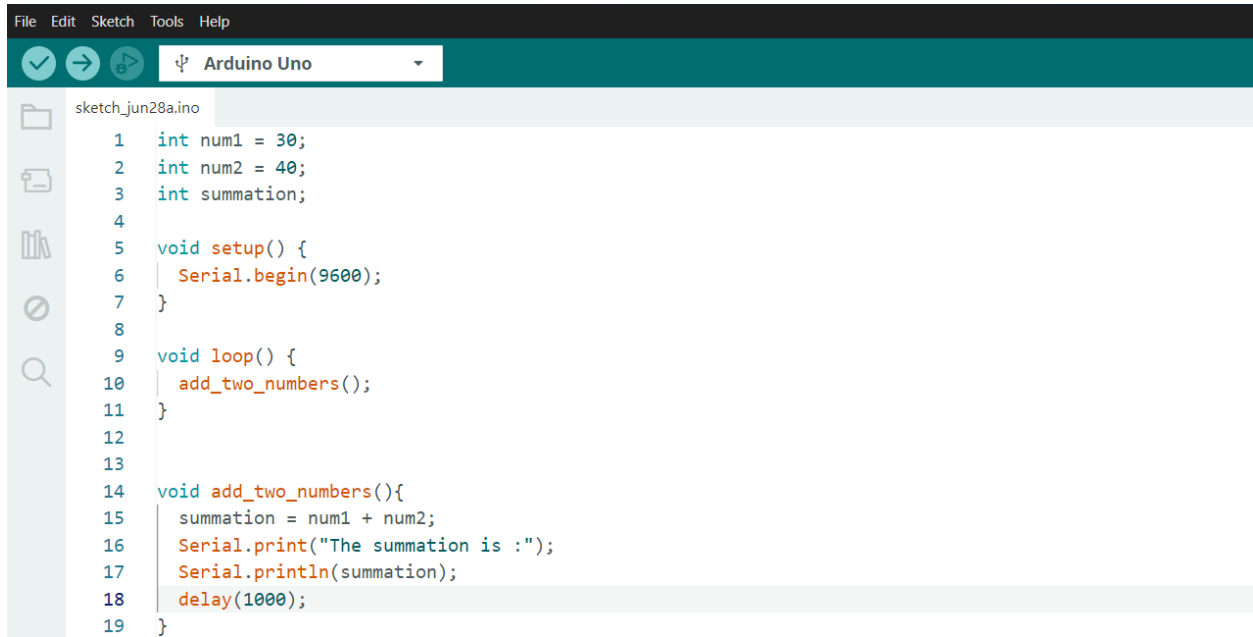
So now anyone can look at your code and say, “Ahha, so this man is trying to turn an LED on and off using this code” because they can understand it by the name LED_on_off().

Let me give you another example of those code for math operation. The code that did addition was:



```
File Edit Sketch Tools Help
1 int num1 = 30;
2 int num2 = 40;
3
4 int summation;
5
6 void setup() {
7   Serial.begin(9600);
8 }
9
10 void loop() {
11   summation = num1 + num2;
12   Serial.print("The summation is: ");
13   Serial.println(summation);
14   delay(1000);
15 }
```

But you can write this code in a more beautiful way. Look at the same code below:



```
File Edit Sketch Tools Help
✓ → ⏮ Arduino Uno
sketch_jun28a.ino
1 int num1 = 30;
2 int num2 = 40;
3 int summation;
4
5 void setup() {
6   Serial.begin(9600);
7 }
8
9 void loop() {
10  add_two_numbers();
11 }
12
13
14 void add_two_numbers(){
15   summation = num1 + num2;
16   Serial.print("The summation is :");
17   Serial.println(summation);
18   delay(1000);
19 }
```

Assignment 03:

1. Produce a music using the buzzer.
2. Make an LED on and off using any pin you like. But at the same time buzzer will give a beep sound. (Beep,LED on...a little delay...beep,LED off...a little delay)
3. Print details of your favourite aircraft/robot in the serial monitor. If you don't have a favourite one find one from google.
4. Practice all the code in Math, condition and loop sections.
5. Print 1 to 500 in the serial monitor using for loop.
6. Print 1 to 500 using while loop.
7. Print 100 to 200 using for loop.
8. Print 100 to 200 using while loop.
9. Print 300 to 100 using for loop.
10. Print 300 to 100 using while loop.
11. Write 0 on SSD but use a user defined function so that other can understand easily.
12. Write 0, then off, then 1, then off, then 2, then off, then 3...thus upto 9. As of yesterday's task but today you have to use a user defined function so that code becomes more readable.
13. Write 0, 1, 2, 3, 4 on the display but with the digit change the buzzer will make a beep sound.
14. List all those things that you didn't realise at all in this PDF (if there are any).

Enough printing for today.