# Task 1

## Q1. Briefly describe what is the different between the void setup (), the void loop () and the main () function.

Void setup() and void loop() are special functions specific to arduino. main() is a part of the C and C++ programming language which is what the arduino superset is based on. When the program runs for the first time, the compiler looks for main() and executes the code within that function. However, we as programmers do not have access to this, instead we have loop() and setup() . Here is what the main function looks like.

int main**(**void**){**

init**();**

#if defined(USBCON)

USBDevice**.**attach**();**

#end if

setup**();**

**for(;;){**

loop**();**

**if** **(**serialEventRun**)** serialEventRun**();**

**}**

**return** 0**;**

**}**

The main function runs setup() once at the start and then loops through loop() indefinitely. Therefore, we use setup() to initialise parts of our program and use loop() for the main program that will need to run continuously.

## Q2. Write a program that will display your name and the ID number once in the serial monitor

void setup**()** **{**

Serial**.**begin**(**9600**);**

Serial**.**println**(**"Filippos Batistatos - 102450186"**);**

**}**

void loop**()** **{**

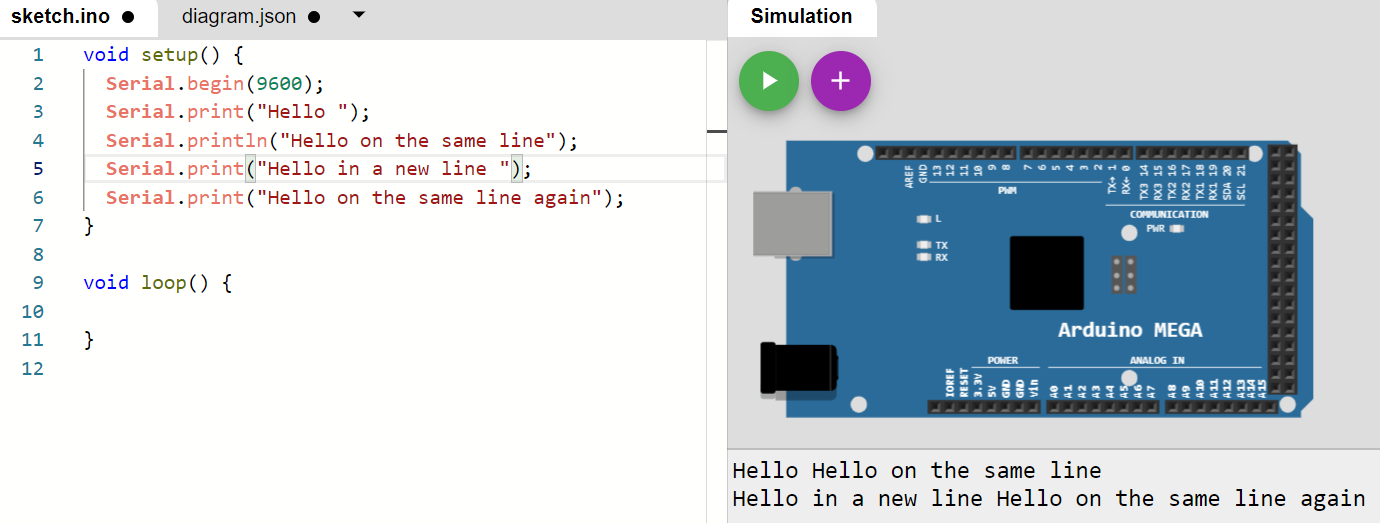
**}**

Graphical user interface, application

Description automatically generated

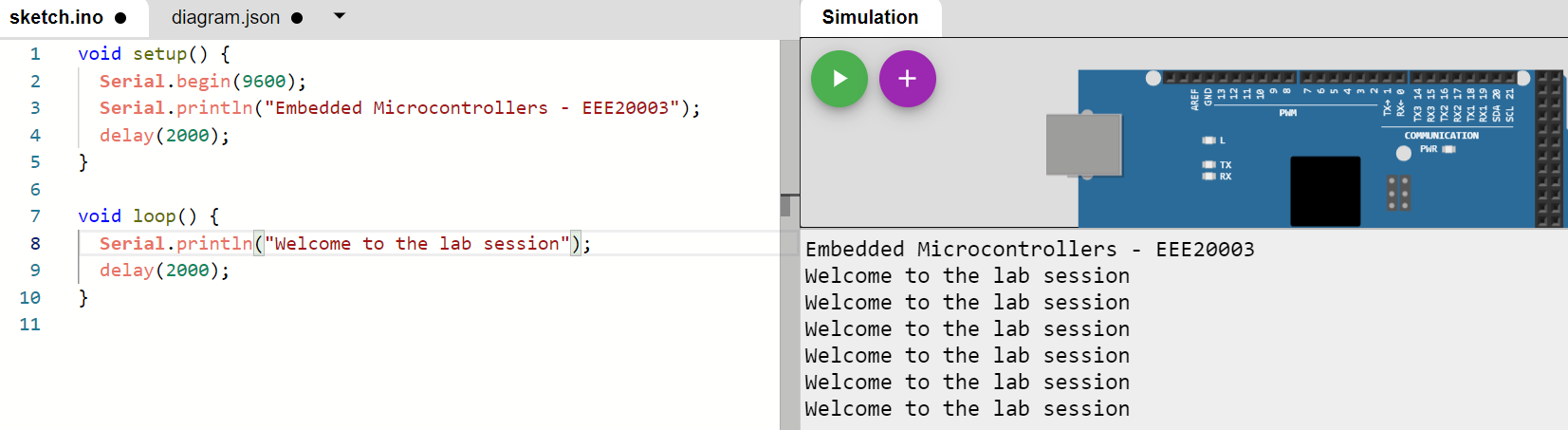
## Q3. Briefly describe the functions delay (),Serial.println(), Serial.begin(), pinMode().

* delay() essentially pauses the code for the number of milliseconds present in the brackets. Delay halts the entire controller allowing it to do nothing else but wait during it’s execution.
* Serial.println() Prints the string placed between the brackets to the serial terminal. Unlike Serial.print(), Serial.println() adds a \n character at the end of the string moving the cursor to the next line.



* Serial.begin() sets the data rate in bits per second - baud - for the serial communication on the board. It is essential when using serial communication that we specify the rate.
* pinMode() sets a pin to a particular mode, input - where we set the value of the pin - or output - where an external device sets the value of the pin and we will need to read that value. This is to ensure we do not have floating values in pins that are important for the operation of our system.

## Q4.Write a program that will display the unit number and the unit’s name at the beginning of the application and then repeatedly print (“Welcome to the lab session”) in every 2 seconds. (Require Demonstration)



## Q5. Write a program that will blink the inbuilt LED every 1 second

void setup**()** **{**

pinMode**(**13**,** OUTPUT**);**

**}**

void loop**()** **{**

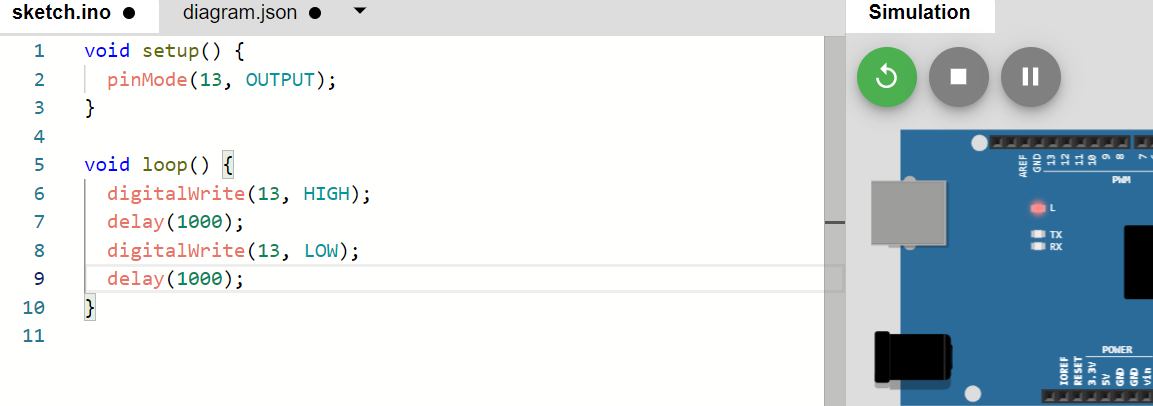
digitalWrite**(**13**,** HIGH**);**

delay**(**1000**);**

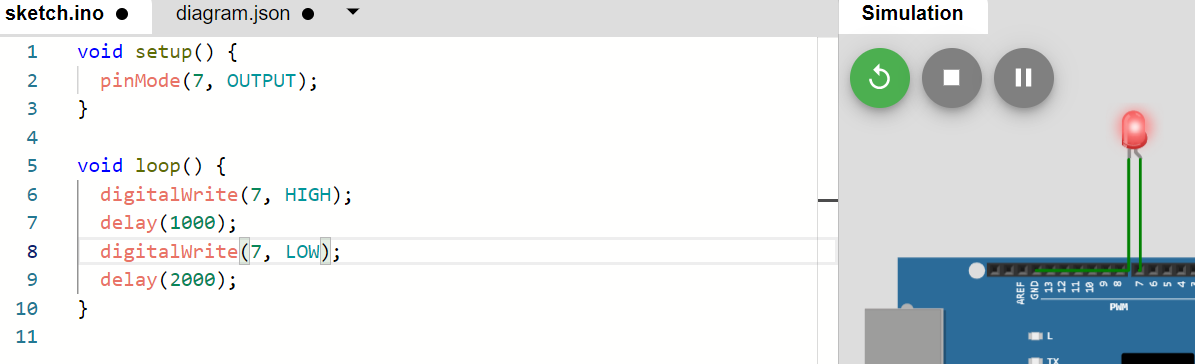
digitalWrite**(**13**,** LOW**);**

delay**(**1000**);**

**}**



## Q6. Write a program that will turn On the LED for 1 second and off the LED for 2 seconds. Make sure that the LED is connected to pin 7. (Require Demonstration)



## Q7. Briefly explain the common data types used in Arduino

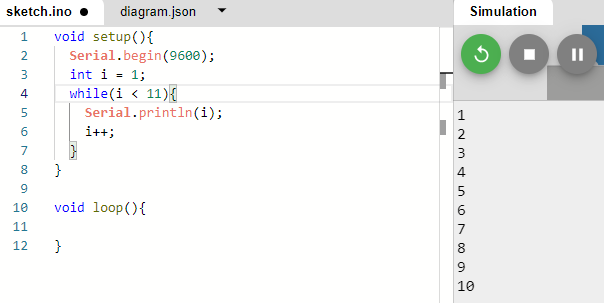
The datatype declaration is important to tell the arduino how much space the data will take up

* boolean it initialises one byte of memory which can be set to wither true or false, 1 and 0 respectively.
* char it initialises one byte of storage to store a character. For single characters we use single quotes, eg. ‘a’, where as for strings we use double quotes eg. “Embedded”. These characters are saved as numbers. Refer to ASCII charts for representation of characters as numbers.
* unsigned char occupies one byte of data, it encodes numbers from 0 to 255
* byte 8 bit unsigned number used to store byte data
* int is the most common way to store integer numbers. It uses 16 bits - 2 bytes - of data and allows for values between -32768 to 32767. There are differences between boards, Arduino Due uses 4 bytes instead of 2.
* unsigned int is similar to int however the value is always positive. This allows for values between 0 and 65535.
* word is used to store a 16 bit unsigned number on the UNO and ATMEGA boards and 32 bit unsigned numbers on the Due and Zero boards.
* long is an extended size variable or number storage. It stores a 32 bit number between -2147483648 and 2147483647.
* unsigned long similar to long but only the positive values allowing for storage between 0 and 2^{32} – 1
* short is a 16 bit data type which leads the range from -2^{15} to 2^{15}-1
* float stores numbers with decimals points. Analog and continuous values are best stored as floats. They are stored in 32 bits of memory and their range varies from -3.4028235\*10^{38} and 3.4028235\*10^{38}
* double is similar to the float datatype. It takes up 32 bits of storage on the arduino Uno and ATMEGA and 64 bits for the arduino Due.

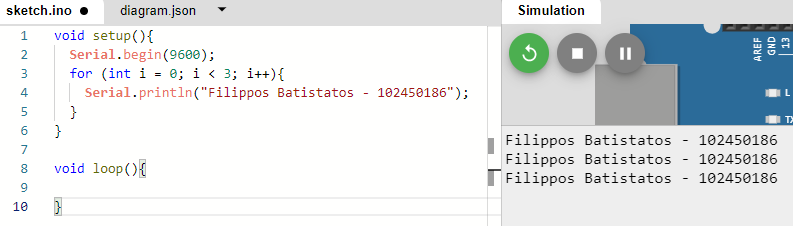
## Q8. Write a simple program to find out the students mark (x) belongs to Higher Distinction, Distinction, Credit, Pass or fail. Add the screenshot for the following marks x= 45,55,60,80 (please use switch statements)(Require Demonstration)



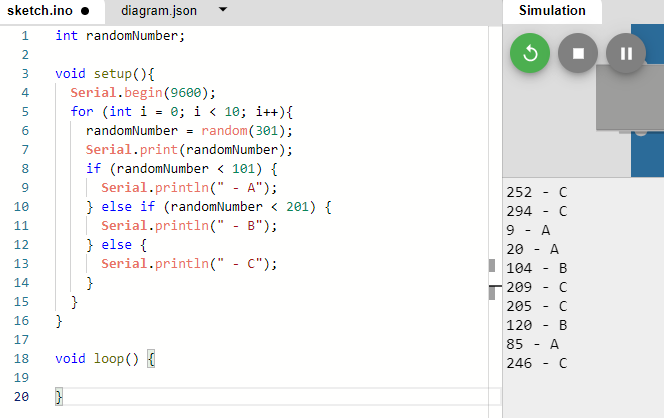
## Q9. Write a simple program that will print the values from 1 to 10 in the serial monitor using a while loop.(Require Demonstration)



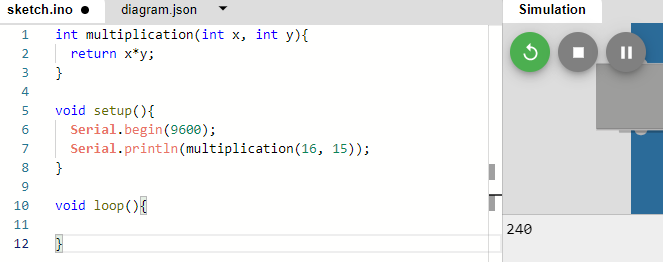
## Q10.Write a program that will print your name and the student ID three times in the serial monitor (Use for loop)(Require Demonstration)



## Q11. Write a program that will generate a random number under 300 for ten times and if the generated number is between 0-100 print A, if the generate number is 101-200 print Band if the generated number is 201-300 print C in the serial monitor. (Use if else if statements and for loop)(Require Demonstration)



## Q12 Write a function to for multiplication, Values needed to multiply be the inputs to the function and function should output the correct value. using the function multiply 16 x 15 and print the correct value in the serial monitor. (Require Demonstration)



# Task 2

## Q1. Connect the Arduino, write a program that will display your name and the ID number once in the serial monitor.

void setup**()** **{**

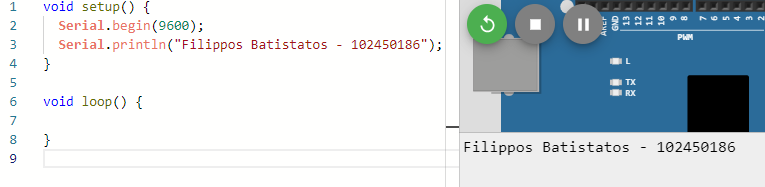
Serial**.**begin**(**9600**);**

Serial**.**println**(**"Filippos Batistatos - 102450186"**);**

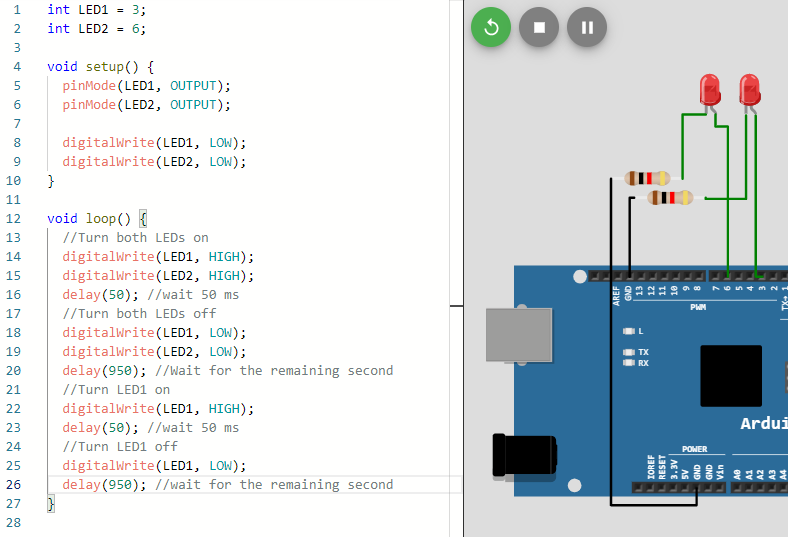
**}**

void loop**()** **{**

**}**

****

## Q2. Connect 2 LEDs (remember to use 270 Ohm resistors). Write code to flash LED1 every 1 second and LED2 every 2 seconds. Flashing LED duration is 50 ms.



int LED1 **=** 3**;** // Positive pin for LED 1

int LED2 **=** 6**;** // Positive pin for LED 2

void setup**()** **{**

pinMode**(**LED1**,** OUTPUT**);**

pinMode**(**LED2**,** OUTPUT**);**

digitalWrite**(**LED1**,** LOW**);**

digitalWrite**(**LED2**,** LOW**);**

**}**

void loop**()** **{**

//Turn both LEDs on

digitalWrite**(**LED1**,** HIGH**);**

digitalWrite**(**LED2**,** HIGH**);**

delay**(**50**);** //wait 50 ms

//Turn both LEDs off

digitalWrite**(**LED1**,** LOW**);**

digitalWrite**(**LED2**,** LOW**);**

delay**(**950**);** //Wait for the remaining second

//Turn LED1 on

digitalWrite**(**LED1**,** HIGH**);**

delay**(**50**);** //wait 50 ms

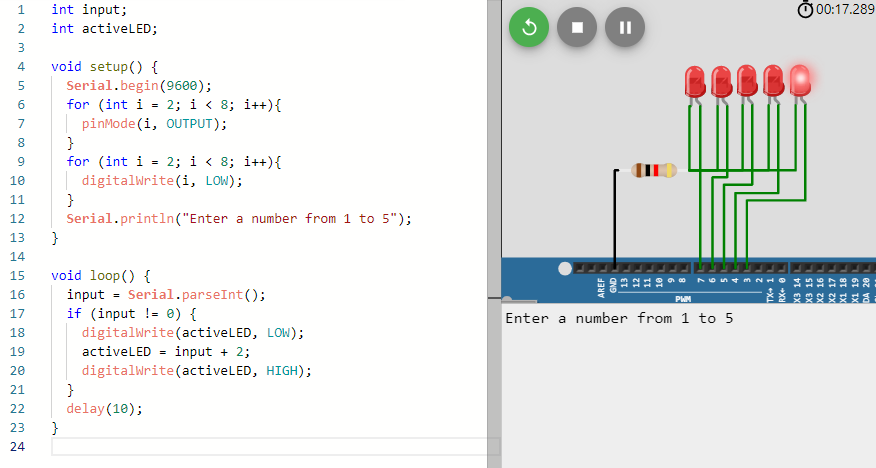
//Turn LED1 off

digitalWrite**(**LED1**,** LOW**);**

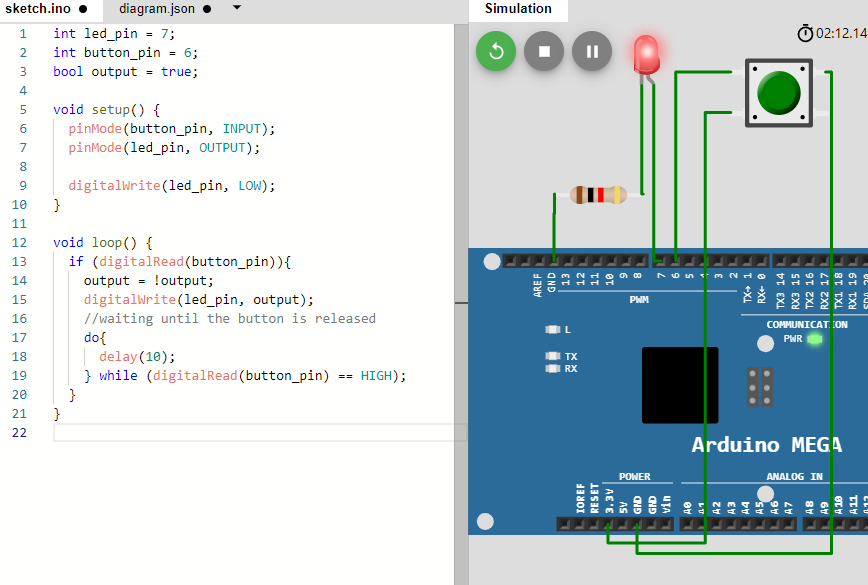
delay**(**950**);** //wait for the remaining second

**}**

## Q3. Connect 5 LEDs (remember to use 270 Ohm resistors). Write code to take input (a single number ranging from 1 to 5) from Serial monitor and light up the corresponding LED. Only one LED lit up at a time. (Require Demonstration)



## Q4. Write code to toggle the LED when the button is pressed. (Require Demonstration)



## Q5. Write code to dim the LED until off and then glow the LED to fully bright (Use PWM).

int led\_pin **=** 7**;**

void setup**()** **{**

pinMode**(**led\_pin**,** OUTPUT**);**

digitalWrite**(**led\_pin**,** LOW**);**

**}**

void loop**()** **{**

**for** **(**int i **=** 0**;** i **<** 255**;** i**++){**

analogWrite**(**led\_pin**,** i**);**

delay**(**10**);**

**}**

**for** **(**int i **=** 255**;** i **>** 0**;** i**--){**

analogWrite**(**led\_pin**,** i**);**

delay**(**10**);**

**}**

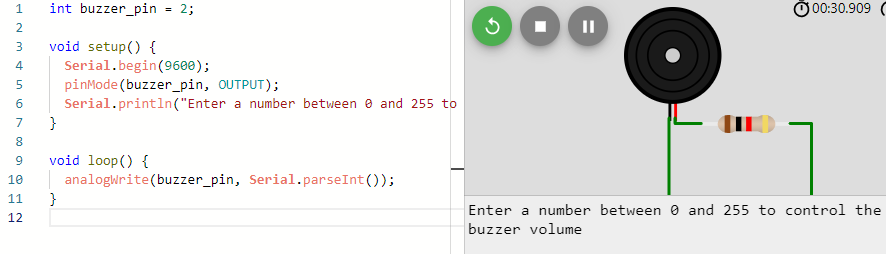
**}**

Graphical user interface, application

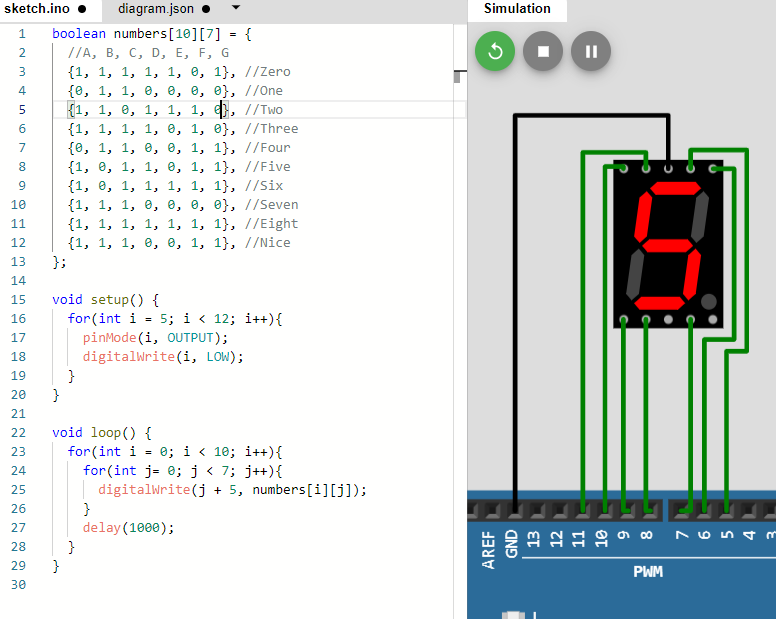
Description automatically generated

\*Note: This tasks does not work particularly well on the simulator. However, when using real hardware, there is a clear difference in brightness as the PWM signal becomes longer.

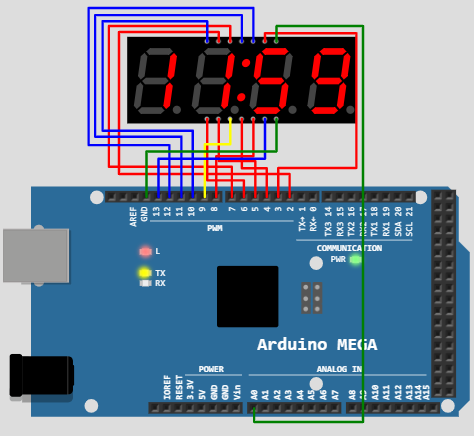
## Q6. Write code to change the volume of the piezo buzzer based on input from Serial monitor (Use PWM).(Require Demonstration)



## Q7. Connect the seven segment LED display, write code to display number from 9 to 0 (change every 1 second). (Require Demonstration)

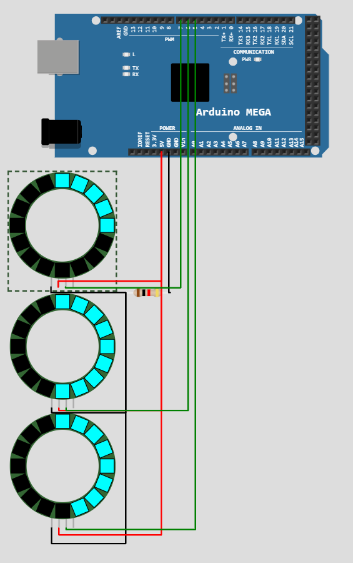


## Q8. Connect the 4-digit seven segment LED display, write code to display “11:59”. (Require Demonstration)

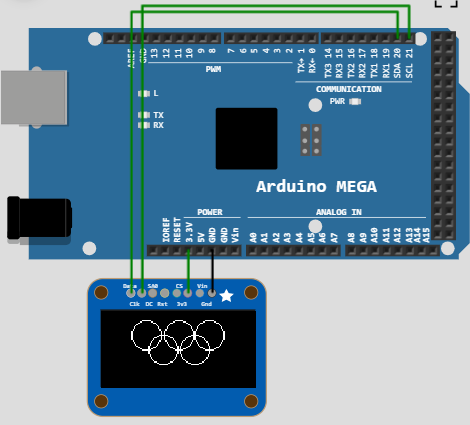


# Task 3

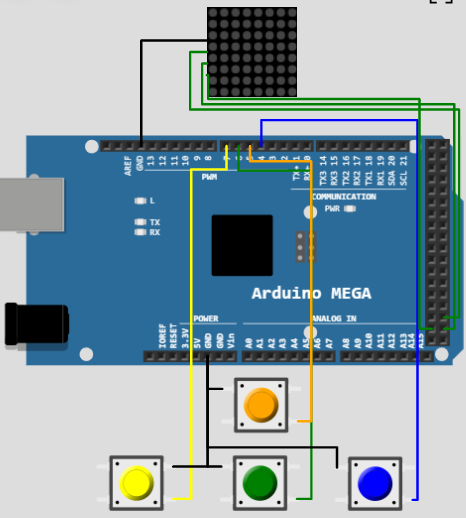
## Q1. Connect 3NeoPixelRings to Arduino Mega. Write code to develop an analog clock display with each NeoPixel Rings presents hour, minute and second. (require demonstration)



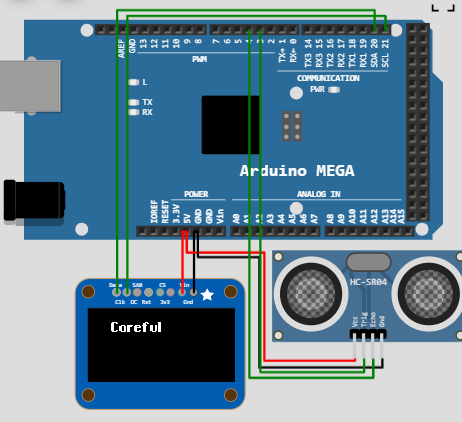
## Q2. Connect OLED display to Arduino Mega. Write code to draw Olympic Rings symbol at center of the OLED display. (require demonstration)



## Q3. Connect LED Matrix with “chain” attribute is 1 and 4 buttons with suitable resistors to Arduino Mega. Write a code to control a dot/LED move up, down, left, and right using 4 buttons. At the beginning, the dot location will be (0,0). Only one dot/LED is displayed at once and this dot/LED can only display within the 8x8 range. (require demonstration)



## Q4. Connect OLED display and HC-SR04 Ultrasonic Distance Sensor to Arduino Mega. Write a code to present the distance value from Ultrasonic sensor as a rectangle (height of the rectangle will be the same while the width of the rectangle will be changed based on the distance value). Display a certain simple text on OLED display based on the distance value: “Safe” when the distance >= 200, “Careful” when 80 <= distance < 200 and “Too Closed” when distance < 80. (require demonstration)

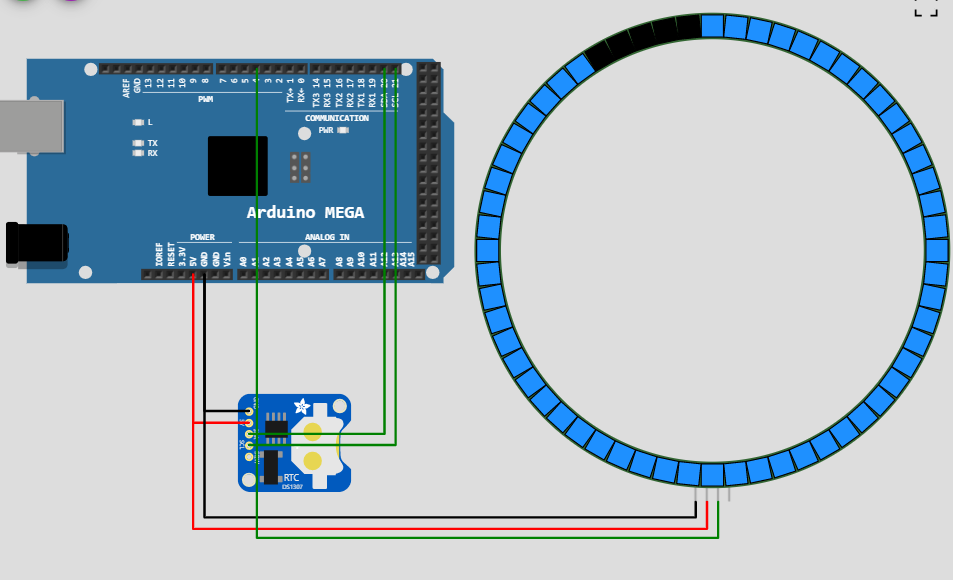


## Q5. Connect DS1307 RTC and LED Matrix with “chain” attribute is 5 to Arduino Mega. Write a code to display the current time from DS1307 RTC to LED Matrix with the speed value is 10, pause value is 100, and text alignment is PA\_CENTER. (require demonstration)

A picture containing text, electronics

Description automatically generated

## Q6. Connect NeoPixel Rings and DS1307 RTC to Arduino Mega. Write a code to create a simple timer using DS1307 RTC and NeoPixel Rings as visualization. In the setup(), write code to take input from serial monitor to set up the hour, minute and second for the timer. (require demonstration)



# Task 4

## Q1. Convert the following numbers to binary, octal, and hexadecimal form: a) 1610, b) 7110.

1610 **:{**

0b**:** 11001001010

Octal**:** 3112

HEX**:** 64A

**}**

7110 **:{**

0b**:** 1101111000110

Octal**:** 15706

HEX**:** 1BC6

**}**

## Q2. Convert the following numbers to the binary form: a) 178, b) FB116.

178**:** 0b**:** 10110010

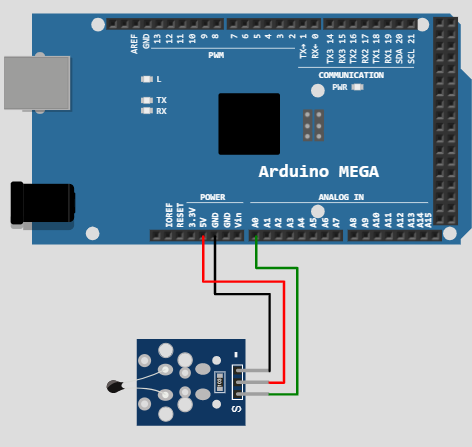
FB116**:** 0b**:** 11111011000100010110

## Q3. Write a program that displays the correct binary representation of a user defined decimal integer between 0 and 255 on the serial monitor. (Do not use built-in functions for this task)(Require demonstration)

A picture containing text, electronics, screenshot

Description automatically generated

## Q4. Write a program that displays the correct reading of the temperature in the immediate surrounding of the temperature sensor in Celsius and Fahrenheit using appropriate calibration steps on the serial monitor.(Require demonstration)



## Q5. Write a program that displays the correct voltage reading from the potentiometer as you dial it on the serial monitor. (Require demonstration)

Graphical user interface

Description automatically generated

## Q6. Write a program that controls the volume of a noise from the speaker using input from the potentiometer.(Require demonstration)

Graphical user interface, diagram

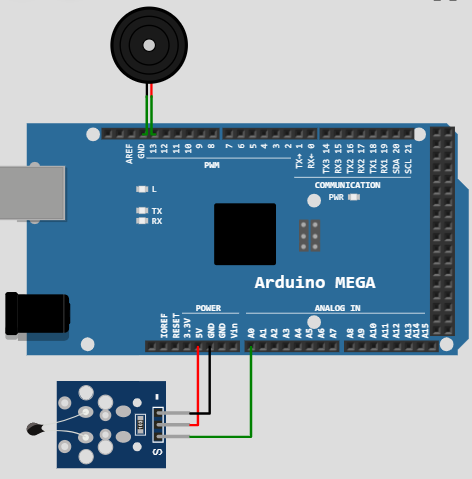
Description automatically generated

## Q7. Write a program that randomly takes an integer from {0,1,2,3} where each randomly taken integer corresponds to an index within the array of LED pins {9,10,11,12}: num “0”corresponds to LED pin 9, num “1”corresponds to LED pin 10 and so on. You are required to generate a specific tune with a single pitch and turn on the right LED if the potentiometer is dialled in the correct quadrant that corresponds to the randomly taken integer value: num “0” also corresponds to a reading between 0% and 25% of the maximum potential, num “1” also corresponds to a reading between 25% and 50% of the maximum potential, and so on. If your dialling on the potentiometer is incorrect, then no LED is lit, and you are required to design a tune of your choice consisting of varying pitches and send it out. Make sure that every tune only lasts for a few seconds and that the outcome of whether you have correctly dialled the potentiometer is displayed on the serial monitor. The game will then be paused for 3 seconds, and a new round will begin. (Require demonstration)

A picture containing text, electronics, screenshot

Description automatically generated

## Q8. Write a program that measures the temperature using a temperature sensor every 5 seconds while simultaneously generating a tune with varying pitches every 2 seconds from the speaker. You are required to display values of temperature at each time of measurement and the text “Speaker is on!” right before a tune is generated and the text “Speaker if off!” right after the tune is turned off. Hint: Use millis() function and a “current time” versus “previous time” structure. (Require demonstration)



# Task 5

## Q1. How many bytes are required to display the string a). “Hello World!”, b). “12”, and c). “My name is: ”.

1. Hello world: 13, 12 for characters 1 for terminating character
2. 12: 3, 2 for characters 1 for terminating character
3. My name is: 13, 12 for characters 1 for terminating character

## Q2. Write a program that returns the byte-wise information of the user input from the Serial monitor in the hexadecimal representation.

void setup**()** **{**

Serial**.**begin**(**9600**);**

**}**

void loop**()** **{**

String message**;**

int msg\_len**;**

Serial**.**println**(**"Please enter an input"**);**

**do** **{**

message **=** Serial**.**readString**();**

**}** **while** **(**message **==** ""**);**

msg\_len **=** message**.**length**()** **+** 1**;**

char char\_array**[**msg\_len**];**

message**.**toCharArray**(**char\_array**,** msg\_len**);**

**for** **(**int i **=** 0**;** i**<sizeof(**char\_array**)-**1**;** i**++)**

**{**

Serial**.**print**(**char\_array**[**i**],** HEX**);**//excludes NULL byte

**}**

Serial**.**print**(**"\n"**);**

**}**

Graphical user interface, text

Description automatically generated

## Q3. Create a 40-byte empty array, move the memory from another array with integer number 0-9 into it, and print every single byte in the 40-byte array once.

int emptyArray**[**40**];**

int numberArray**[**10**]** **=** **{**0**,** 1**,** 2**,** 3**,** 4**,** 5**,** 6**,** 7**,** 8**,** 9**};**

void setup**()** **{**

Serial**.**begin**(**9600**);**

emptyArray**[**2**]** **=** numberArray**;**

**for** **(**int i **=** 0**;** i **<** 40**;** i**++){**

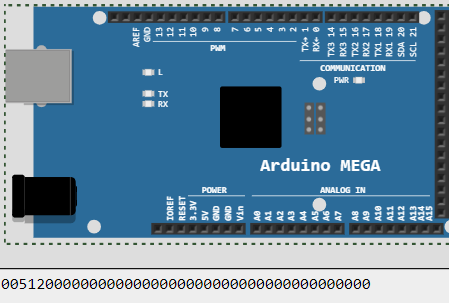
Serial**.**print**(**emptyArray**[**i**]);**

**}**

**}**

void loop**()** **{**

**}**

****

## Q4. Create a 2 by 3 matrix with 1 -3 in the first row and 4-6 in the next. Create a function that can print out the input array in matrix format. Display the matrix on the serial monitor using this function.

int matrix**[**2**][**3**]** **=** **{{**1**,**2**,**3**},{**4**,**5**,**6**}};**

void setup**()** **{**

Serial**.**begin**(**9600**);**

**for(**int i **=** 0**;** i **<** 2**;** i**++){**

**for** **(**int j **=** 0**;** j **<** 3**;** j**++){**

Serial**.**print**(**matrix**[**i**][**j**]);**

Serial**.**print**(**" "**);**

**}**

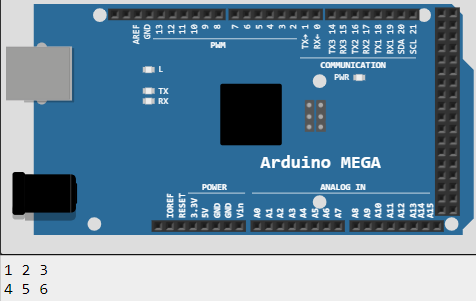
Serial**.**print**(**'\n'**);**

**}**

**}**

void loop**()** **{**

**}**

****

## Q5. How do we access SRAM and flash memory? When do we use SRAM and when do we use flash memory for data storage?

**SRAM** is used to store variables during the execution of the program. This is where the executable program stores all its variables when it is initialisation but also stores new variables.

**Flash Memory** Only used when a very large block of data needs to be stored. Due to it's slower speed it is better to store things that do not need to change frequently but only be read. In order to store things there we must make them constants. This is done by writing const before the variable name.

## Q6. Use the built-in utility PROGMEM from the library pgmspace.h to store a string “ARDUINO IS FUN, I WANT TO DO MORE.” and a character array with numbers 6500, 2300, 20, 1, 2, 14, and 5 in the flash memory of Arduino DUE instead of RAM. On the serial monitor, please print the string followed by an empty line, and each number form the character array on a separate line only once. (Require demonstration)

Graphical user interface

Description automatically generated

## Q7. Create a program that turns on and off individual LEDs using the middle 8 LEDs from an LED bar graph using the numeric input from the serial monitor. You need to use an array to initialise all digital input pins and create a void function that prints out which LED (from 1 to 8) is being turned on e.g., ‘OUTPUT X ON’, turns up the brightness of the selected LED, and then turns down its brightness for an appropriate duration of time and prints ‘OUTPUT X OFF’. It is recommended that you use the network resistor and LED bar graph for this exercise. (Require demonstration)

A picture containing diagram

Description automatically generated

# Task 6

## Q1.Briefly explain about the Timer Counters in Arduino Mega and How to calculate the time using the Timer counters.

millis() returns the time in milliseconds since the program started running.

By measuring the time elapsed we can create a timer. This is as simple as capturing the time when the program starts and then capturing it once more later on and comparing the two times. When the time difference between the two values is equal or larger than the amount you need to count we can trip a flag or turn on an LED to indicate that the time has ellapsed.

## Q2. Write a simple program to blink the LED every 500 milliseconds. When the push button is pressed the LED blinking frequency should be double. (use hardware interrupts)

int LEDPin **=** 3**;**

int buttonPin **=** 6**;**

int previousMillis**;**

int currentMillis**;**

int frequency **=** 500**;** //frequency in milliseconds

void setup**()** **{**

pinMode**(**LEDPin**,** OUTPUT**);**

pinMode**(**4**,** OUTPUT**);**

pinMode**(**buttonPin**,** INPUT**);**

//setting the negative LED pin

digitalWrite**(**4**,** LOW**);**

previousMillis **=** millis**();**

**}**

void loop**()** **{**

blinkCheck**();**

**if** **(**digitalRead**(**buttonPin**)** **==** HIGH**){**

frequency **=** frequency **/** 2**;**

**do{**

delay**(**1**);**

blinkCheck**();**

**}while(**digitalRead**(**buttonPin**));**

**}**

**}**

void blinkCheck**(){**

currentMillis **=** millis**();**

**if** **(**currentMillis **-** previousMillis **>=** frequency**){**

digitalWrite**(**LEDPin**,** HIGH**);**

delay**(**20**);**

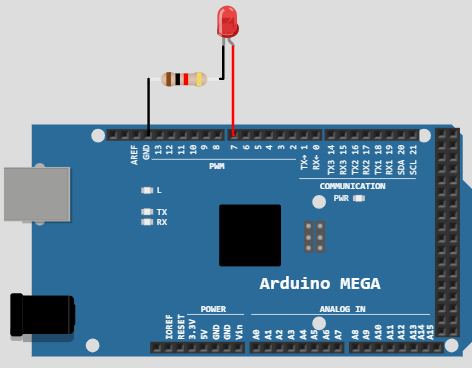
digitalWrite**(**LEDPin**,** LOW**);**

previousMillis **=** currentMillis**;**

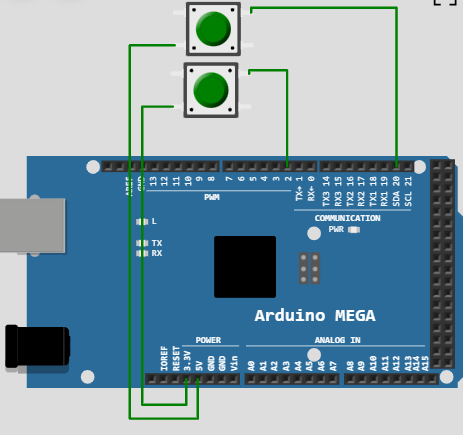
**}**

**}**

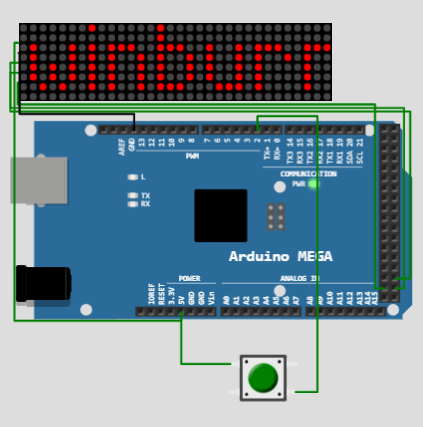
## Q3.Write a simple program to blink the LED every 5 second using timer interrupts (Do not use delay, millis, micros ) . (Require demonstration)



## Q4. Write a simple program to blink the inbuilt LED every 500 milliseconds. Connect two buttons to the Arduino (Button A & Button B) When A is pressed millis value should be printed in the serial monitor and when the B is pressed micros value should be printed in the serial monitor. (Use hardware interrupts) . (Require demonstration)



## Q5. Write a program to display “I love Swinburne” in the LED matrix. When the external button is pressed it should display your name. The text should move from left to right continuously.(Use LED Dot Matrix MA7219 and interrupts).(Require demonstration)



## Pass Plus Tasks

## Q1. To design digital clock with few features. Connect OLED display to Arduino. Write program to display analog and digital clocks. At the beginning, the clock is set up using serial. A real time clock (RTC) module can be added as well.(Require demonstration)

A picture containing text, electronics

Description automatically generated

## Q2. To design LED matrix with few features. Make a proper connection for 8×8 LED Matrix with resistors to the microcontroller. The LED matrix have to be constructed manually by using 64 units of LED as 8×8 LED Matrix. Display your name andscroll from right to left continuously.. (Require demonstration)

Shape, arrow

Description automatically generated

# Credit Tasks

## Q1.Continuing from Pass Plus Task 1 for designing digital clock with few features. Connect to a push button. Use the hardware interrupt to detect the button to set/change the clock setting using single button.. (Require demonstration)

Diagram

Description automatically generated

## Q2. Continuing from Pass Plus Task 2 for designing LED matrix with few features. Use the serial for allowing change the text of the display and save it to memory(RAM/EEPROM).(Require demonstration)

A picture containing diagram

Description automatically generated

# Distinction Tasks

## Q1.Continuing from Credit Task 1 for designing digital clock with few features. Using the same button to have additional feature to set alarm system. Connect to a speaker for the output of the alarm.. (Require demonstration)

Graphical user interface, diagram

Description automatically generated

## Q2. Continuing from Credit Task 2 for designing LED matric with a few features. Connect the pushbutton to the embedded system. Add feature with the button. If button press for 1 second or less, the text will moving from left to right. If the button is pressed for more than 2 seconds, the text will move from right to left. (Require demonstration)

Diagram

Description automatically generated

# High Distinction

## Q1. Continuing from Distinction Task 1 for designing a digital clock with a few features. Connect the temperature sensor to the microcontroller. Using previous button to have additional feature to turn on/off displaying temperature on the OLED display. If it is activated, read temperature sensor and display on OLED display. (Require demonstration)

Diagram, schematic

Description automatically generated

## Q2. Continuing from Credit task 2 for designing LED matrix with a few features. Additional features is need to be added with the same pushbutton to change up to 3 modes for speed of the moving text includes slow, medium, fast. (Require demonstration)

Diagram

Description automatically generated with medium confidence