CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY AND RESEARCH

NAME: Parth N Patel

ID: 19DCS098

SUBJECT: MPCO

SEM:4

PRACTICAL-1

AIM:

a) Write an ALP to move block of data bytes from one location to another location.

b) Write an ALP to exchange block of data bytes.

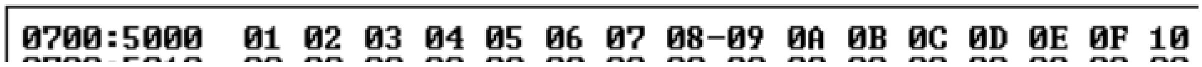
PROGRAM CODE:

(a) Moving Block of data

```
org 100h

mov [5000h],1
mov [5001h],2
mov [5002h],3
mov [5003h],4
mov [5004h],5
mov [5005h],6
mov [5006h],7
mov [5007h],8
mov [5008h],9
mov [5009h],10
mov [500Ah],11
mov [500Bh],12
mov [500Ch],13
mov [500Dh],14
mov [500Eh],15
mov [500Fh],16


mov cl,16
mov si,5000h
mov di,7000h


l: mov al,[si]
   mov [di],al
   inc si
   inc di
   dec cl
   jnz l
ret
```

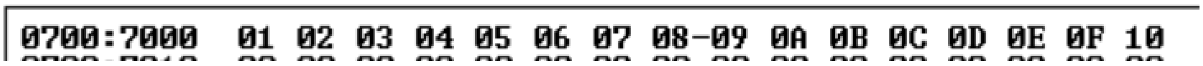
OUTPUT:

Data stored in the location 5000H-500FH



```
0700:5000  01 02 03 04 05 06 07 08-09 0A 0B 0C 0D 0E 0F 10
```

After Execution of Program:



```
0700:7000  01 02 03 04 05 06 07 08-09 0A 0B 0C 0D 0E 0F 10
```

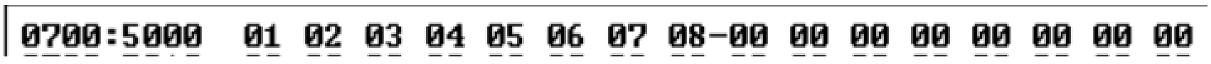
(b)Exchange Block of Data

```
org 100h

mov [5000h],1
mov [5001h],2
mov [5002h],3
mov [5003h],4
mov [5004h],5
mov [5005h],6
mov [5006h],7
mov [5007h],8

mov [7000h],10H
mov [7001h],20H
mov [7002h],30H
mov [7003h],40H
mov [7004h],50H
mov [7005h],60H
mov [7006h],70H
mov [7007h],80H

mov cl,8H
mov si,5000H
mov di,7000H
```

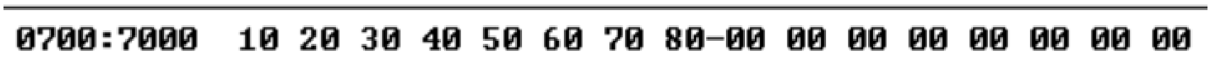
```
l: mov al,[si]
   mov bl,[di]
   mov [si],bl
   mov [di],al
   inc si
   inc di
   dec cl
   jnz l
ret
```

OUTPUT:

BEFORE:

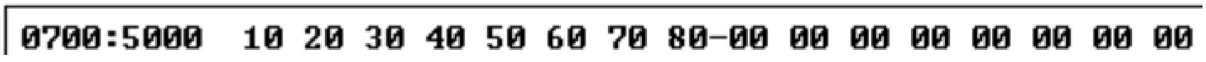


```
0700:5000  01 02 03 04 05 06 07 08-00 00 00 00 00 00 00 00
```

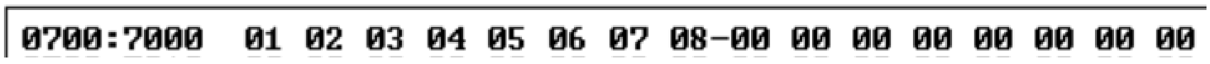


```
0700:7000  10 20 30 40 50 60 70 80-00 00 00 00 00 00 00 00
```

AFTER:



```
0700:5000  10 20 30 40 50 60 70 80-00 00 00 00 00 00 00 00
```



```
0700:7000  01 02 03 04 05 06 07 08-00 00 00 00 00 00 00 00
```

PRACTICAL-2

AIM:

a) Write an ALP to perform 16-bit and 32-bit addition and subtraction.

b) Write an ALP to perform 16-bit and 32-bit multiplication.

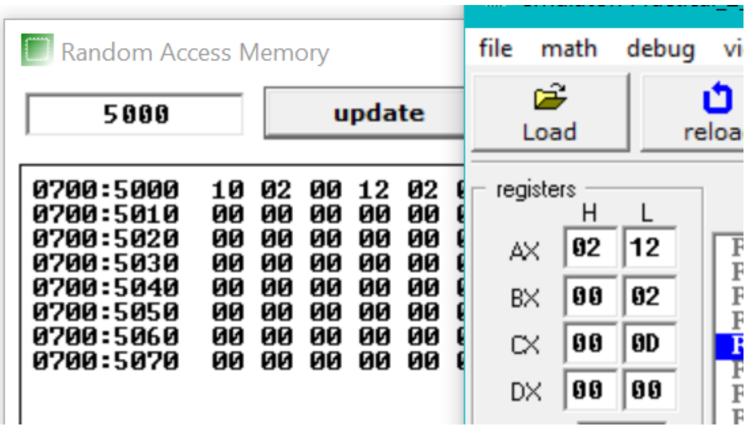
PROGRAM CODE:

(a) 16-bit and 32-bit Addition and Subtraction

16-bit Addition:

```
org 100h
MOV AX , [5000H]
MOV BX , [5001H]

ADD AX, BX
MOV [5003H],AX

Ret
```

OUTPUT:



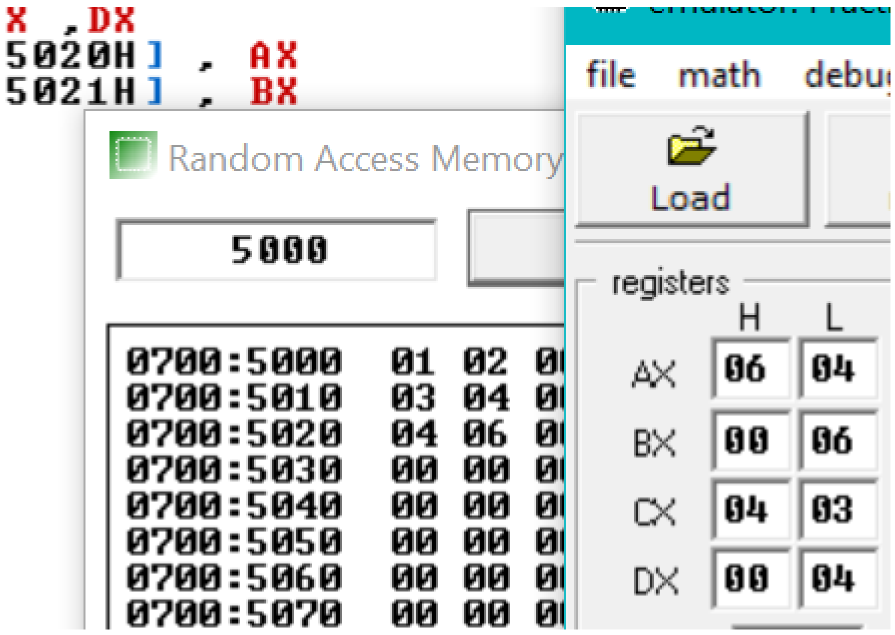
```
5000

0700:5000  10 02 00 12 02
0700:5010  00 00 00 00 00
0700:5020  00 00 00 00 00
0700:5030  00 00 00 00 00
0700:5040  00 00 00 00 00
0700:5050  00 00 00 00 00
0700:5060  00 00 00 00 00
0700:5070  00 00 00 00 00
```

```
     H   L
AX  02  12
BX  00  02
CX  00  0D
DX  00  00
```

32-Bit Addition:

```
org 100h
MOV AX , [5000H]
MOV BX , [5001H]
MOV CX , [5010H]
MOV DX , [5011H]
ADD AX ,CX
ADD BX ,DX
MOV [5020H] , AX
MOV [5021H] , BX
Ret
```

OUTPUT:



```
X  ,DX
5020H] , AX
5021H] , BX
```

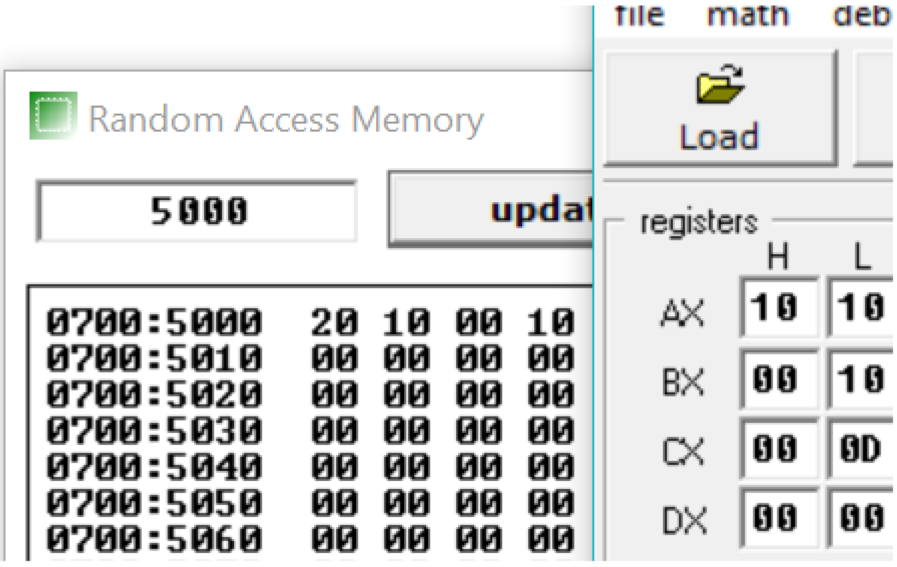
```
0700:5000  01 02 0
0700:5010  03 04 0
0700:5020  04 06 0
0700:5030  00 00 0
0700:5040  00 00 0
0700:5050  00 00 0
0700:5060  00 00 0
0700:5070  00 00 0
```

	H	L
AX	06	04
BX	00	06
CX	04	03
DX	00	04

16-Bit Subtraction:

```
org 100h
MOV AX , [5000H]
MOV BX , [5001H]

SUB AX, BX
MOV [5003H],AX

ret
```

OUTPUT:



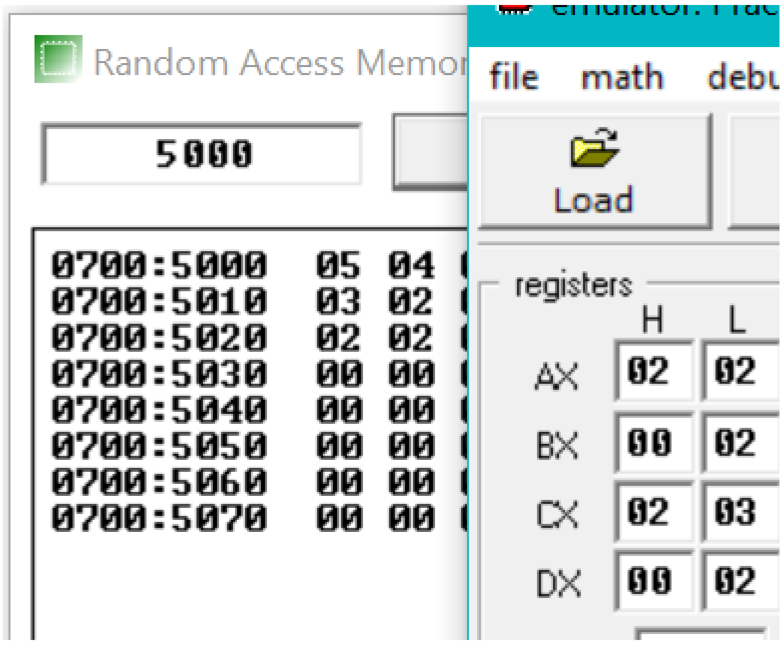
```
0700:5000   20 10 00 10
0700:5010   00 00 00 00
0700:5020   00 00 00 00
0700:5030   00 00 00 00
0700:5040   00 00 00 00
0700:5050   00 00 00 00
0700:5060   00 00 00 00
```

	H	L
AX	10	10
BX	00	10
CX	00	0D
DX	00	00

32-Bit Subtraction:

```
org 100h
MOV AX , [5000H]
MOV BX , [5001H]
MOV CX , [5010H]
MOV DX , [5011H]
SUB AX ,CX
SUB BX ,DX
MOV [5020H] , AX
MOV [5021H] , BX
ret
```

OUTPUT:



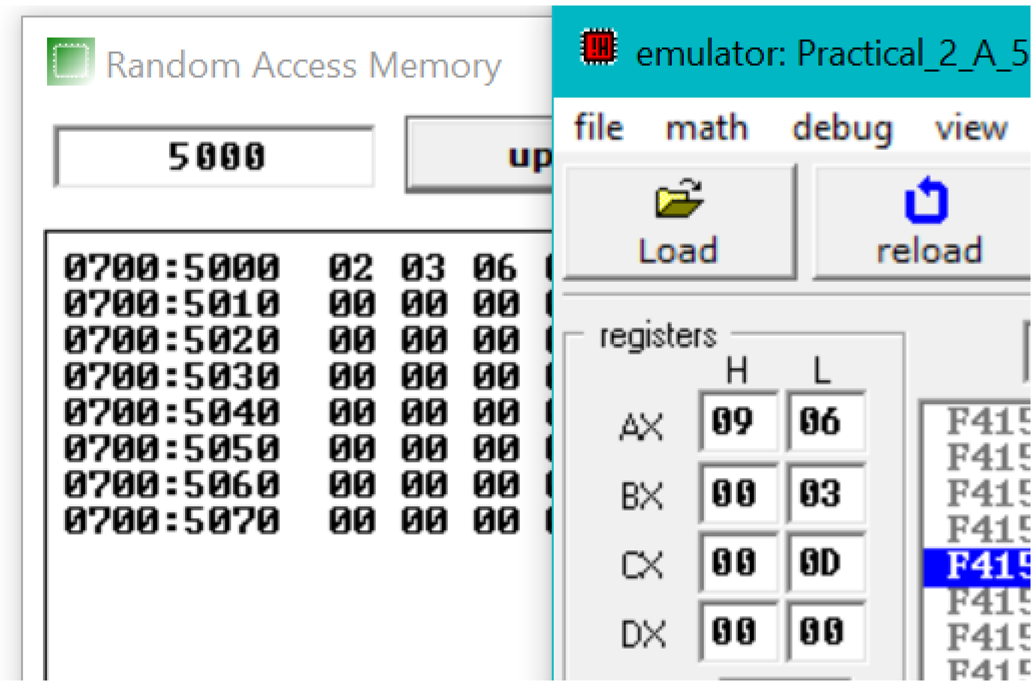
```
5000

0700:5000   05  04
0700:5010   03  02
0700:5020   02  02
0700:5030   00  00
0700:5040   00  00
0700:5050   00  00
0700:5060   00  00
0700:5070   00  00
```

```
     H   L
AX  02  02
BX  00  02
CX  02  03
DX  00  02
```

16-Bit Multiplication:

```
org 100h
MOV AX,[5000H]
MOV BX,[5001H]
MUL BX
MOV [5002H],AX
Ret
```

OUTPUT:



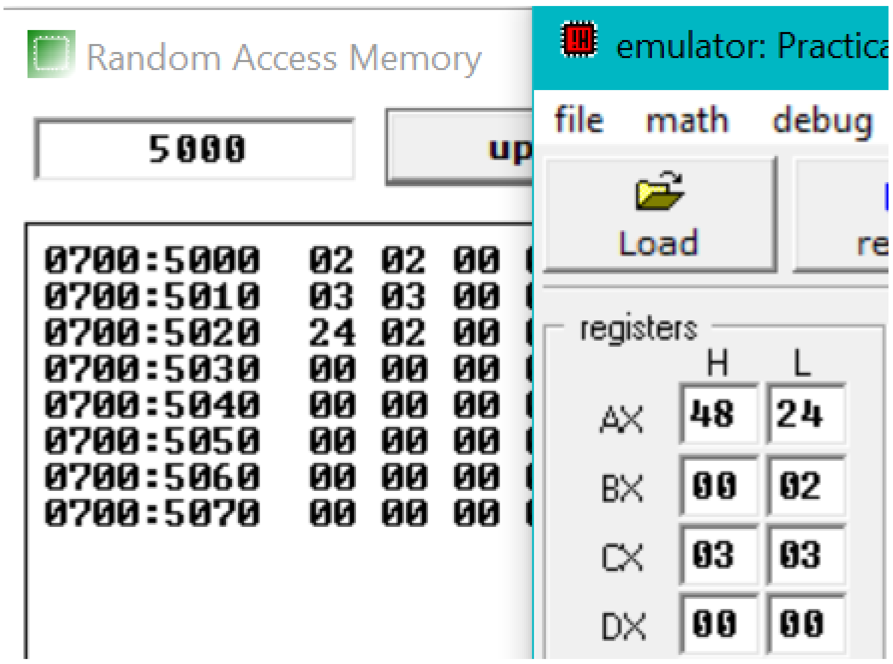
```
0700:5000  02 03 06
0700:5010  00 00 00
0700:5020  00 00 00
0700:5030  00 00 00
0700:5040  00 00 00
0700:5050  00 00 00
0700:5060  00 00 00
0700:5070  00 00 00
```

	H	L
AX	09	06
BX	00	03
CX	00	0D
DX	00	00

32-Bit Multiplication:

```
org 100h
MOV AX , [5000H]
MOV BX , [5001H]
MOV CX , [5010H]
MOV DX , [5011H]
MUL CX
MUL DX
MOV [5020H] , AX
MOV [5021H] , BX
ret
```

OUTPUT:



```
0700:5000  02 02 00
0700:5010  03 03 00
0700:5020  24 02 00
0700:5030  00 00 00
0700:5040  00 00 00
0700:5050  00 00 00
0700:5060  00 00 00
0700:5070  00 00 00
```

```
    H  L
AX  48 24
BX  00 02
CX  03 03
DX  00 00
```

PRACTICAL-3

AIM:

a) Write an ALP to perform sorting of array in ascending order.

b) Write an ALP to perform sorting of array in descending order.

PROGRAM CODE:

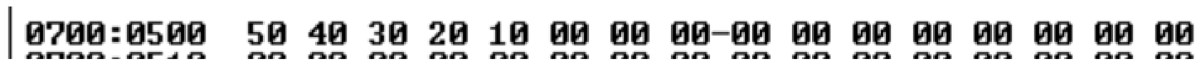
(a) Sorting of array in ascending order

```
 org 100h

mov [500h],50H
mov [501h],40H
mov [502h],30H
mov [503h],20H
mov [504h],10H

mov dl,5
mov si,500h
mov di,500h

l2: mov al,[si]
mov cl,dl

l1: inc si
mov bl,[si]
```

```
cmp al,bl

jz next
jc next

xchg al,bl

mov [si],bl

next: loop l1

mov [di],al
inc di
mov si,di
dec dl

jnz l2

ret
```

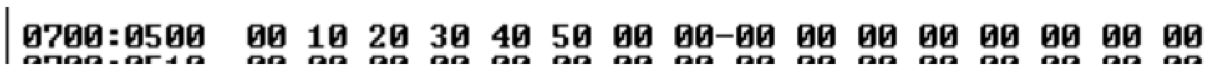
OUTPUT:

BEFORE:



```
0700:0500  50 40 30 20 10 00 00 00-00 00 00 00 00 00 00 00
```

AFTER:



```
0700:0500  00 10 20 30 40 50 00 00-00 00 00 00 00 00 00 00
```

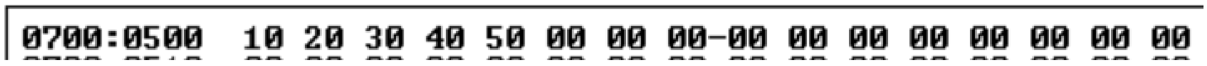
b) Write an ALP to perform sorting of array in descending order.

```
org 100h

mov [500h],10H
mov [501h],20H
mov [502h],30H
mov [503h],40H
mov [504h],50H

mov dl,5
mov si,500h
mov di,500h

l2: mov al,[si]
mov cl,dl

l1: inc si
mov bl,[si]
cmp al,bl

jz next
jnc next

xchg al,bl
```

```
mov [si],bl

next: loop l1

mov [di],al
inc di
mov si,di
dec dl

jnz l2

ret
```

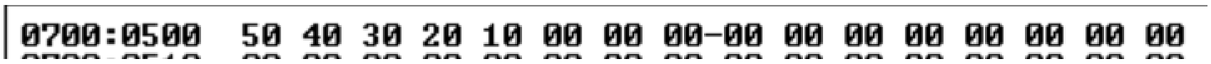
<u>OUTPUT:</u>

BEFORE:



```
0700:0500  10 20 30 40 50 00 00 00-00 00 00 00 00 00 00 00
```

AFTER:



```
0700:0500  50 40 30 20 10 00 00 00-00 00 00 00 00 00 00 00
```

PRACTICAL-4

AIM:

a) Write an ALP to perform factorial of a number

b) Write an ALP to check whether the given 16-bit number is palindrome or not

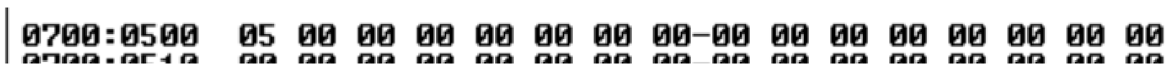
PROGRAM CODE:

a) Write an ALP to perform factorial of a number

```
org 100h


mov [500h],5

mov ax,[500h]

mov bx,ax


dec bx

l1: mul bx

dec bx


jnz l1


mov [500h],ax


ret
```

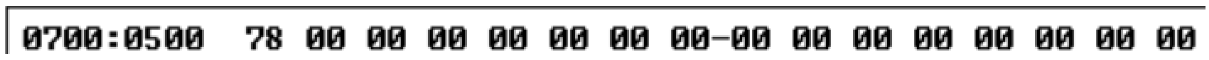
OUTPUT:

BEFORE:



```
0700:0500  05 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

AFTER:



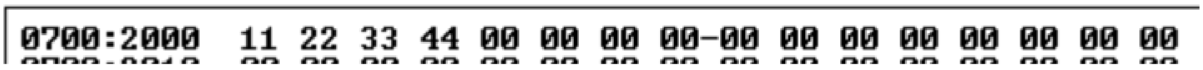
```
0700:0500  78 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

78 is hexadecimal value of 120.

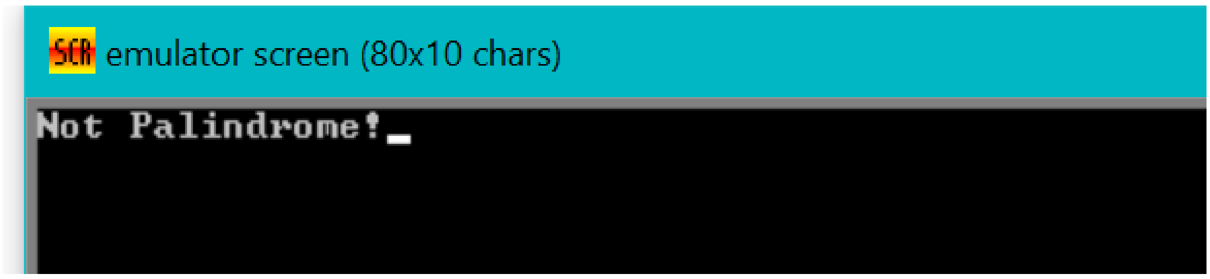
b) Write an ALP to check whether the given 16-bit number is palindrome or not

```
org 100h
MOV SI,2000H
MOV DI,2004H
MOV CL,05H
next_char:
MOV AL,[DI]
MOV BL,[SI]
CMP AL,BL
JNE not_palindrome
INC SI
DEC DI
loop next_char
is_palindrome:
MOV AH,09H
MOV DX,OFFSET msg1
INT 21H
jmp stop
not_palindrome:
MOV AH,09H
MOV DX,OFFSET msg2
INT 21H
stop:
MOV AH,00H
INT 16H

ret
msg1 db "Palindrome!$"
msg2 db "Not Palindrome!$"
```

OUTPUT:



```
0700:2000  11 22 33 44 00 00 00 00-00 00 00 00 00 00 00 00
```



```
Not Palindrome!_
```

Extra Practical :

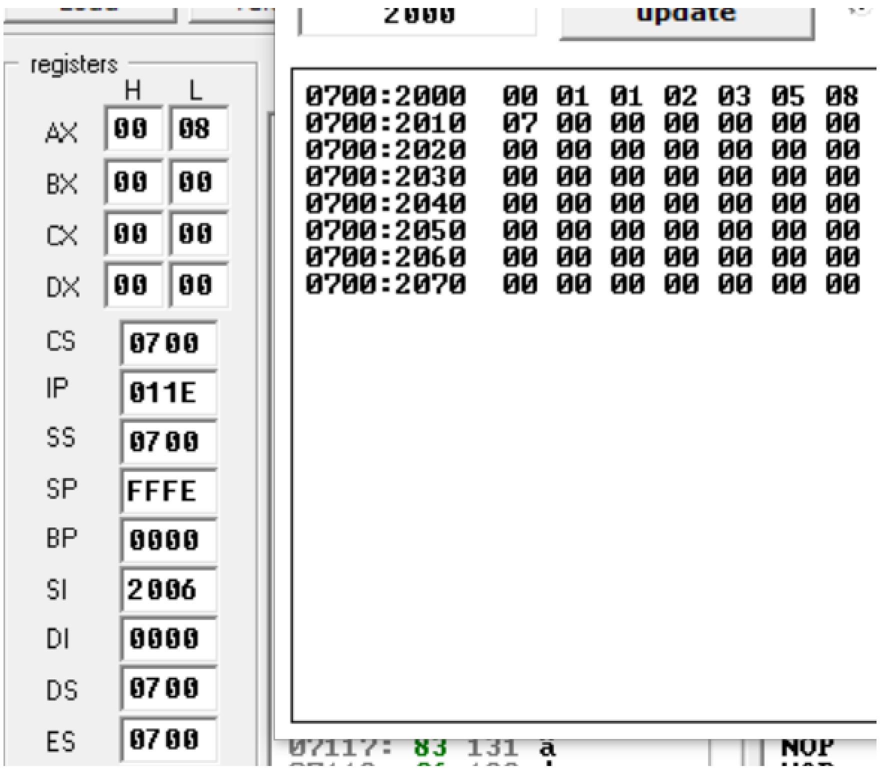
AIM:

Write an ALP to perform Fibonacci series

PROGRAM CODE:

```
org 100h
MOV AL, 00H
MOV SI, 2001H
MOV [SI], AL
ADD AL, 01H
MOV [SI], AL
MOV CX, [2010H]
SUB CX, 0002H
L1: MOV AL, [SI-1]
ADD AL, [SI]
ADD SI, 01H
MOV [SI], AL
LOOP L1
HLT
Ret
```

OUTPUT:



```
AX 00 08
BX 00 00
CX 00 00
DX 00 00
CS 0700
IP 011E
SS 0700
SP FFFE
BP 0000
SI 2006
DI 0000
DS 0700
ES 0700
```

```
0700:2000  00 01 01 02 03 05 08
0700:2010  07 00 00 00 00 00 00
0700:2020  00 00 00 00 00 00 00
0700:2030  00 00 00 00 00 00 00
0700:2040  00 00 00 00 00 00 00
0700:2050  00 00 00 00 00 00 00
0700:2060  00 00 00 00 00 00 00
0700:2070  00 00 00 00 00 00 00
```

```
07117: 83 131 â      NOP
```

PRACTICAL-5

(a) Develop a program to interface Arduino with LED and blink led for 1second.

(b) Develop a program to interface Input Switches and output LEDs with Arduino.

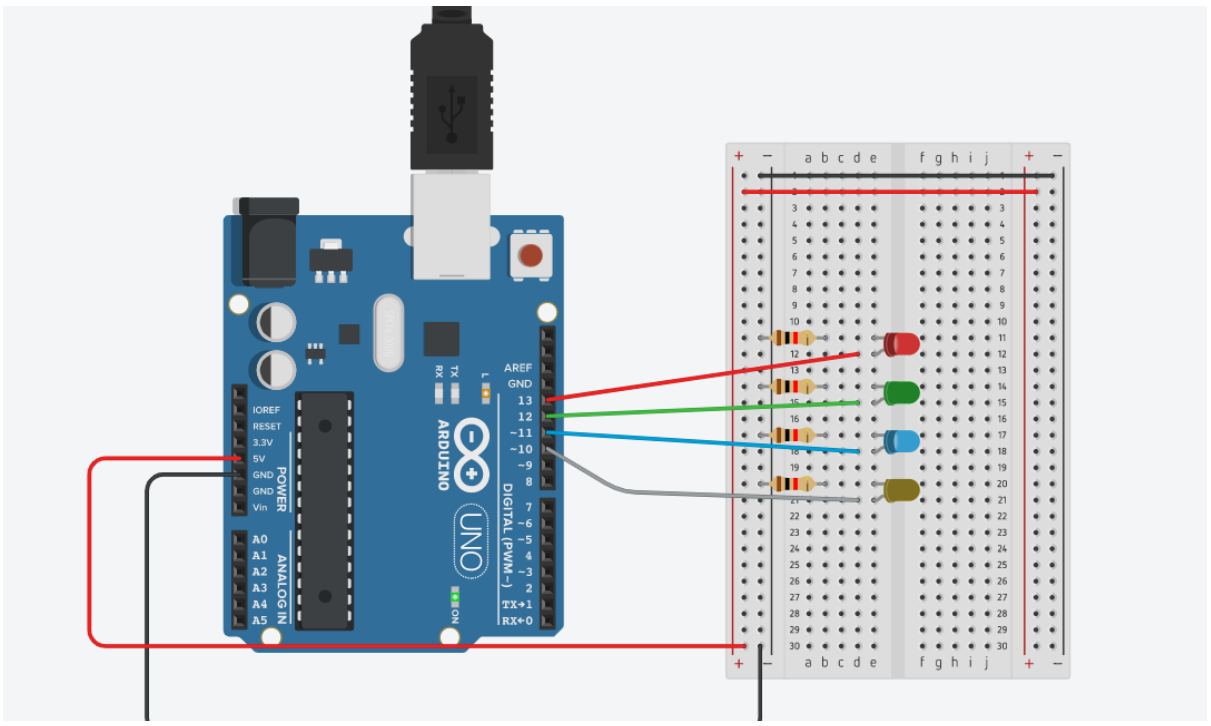
PROGRAM CODE:

(a) Develop a program to interface Arduino with LED and blink led for 1second.

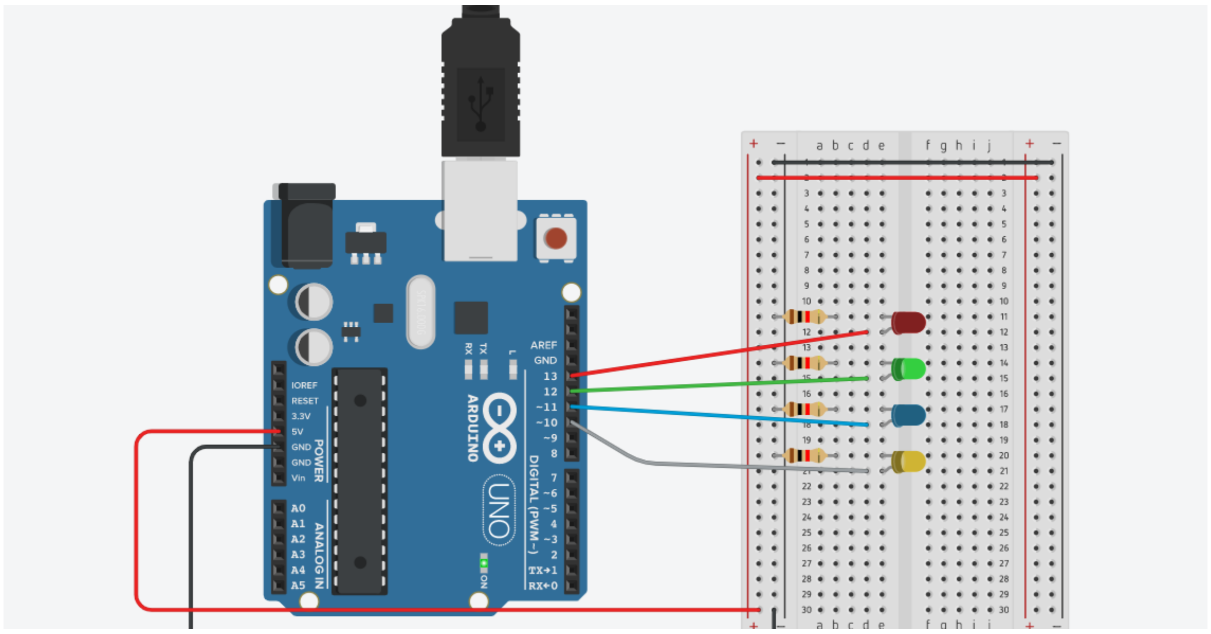
PROGRAM CODE:

```
void setup()
{
  pinMode(13, OUTPUT);
  pinMode(12,OUTPUT);
  pinMode(11,OUTPUT);
  pinMode(10,OUTPUT);


}

void loop()
{
  digitalWrite(12,LOW);
  digitalWrite(10,LOW);
  digitalWrite(13, HIGH);
  digitalWrite(11,HIGH);
  delay(2000);
  digitalWrite(13,LOW);
  digitalWrite(11,LOW);
  digitalWrite(12,HIGH);
  digitalWrite(10,HIGH);
  delay(2000);
}
```

OUTPUT:



Red and Blue light on



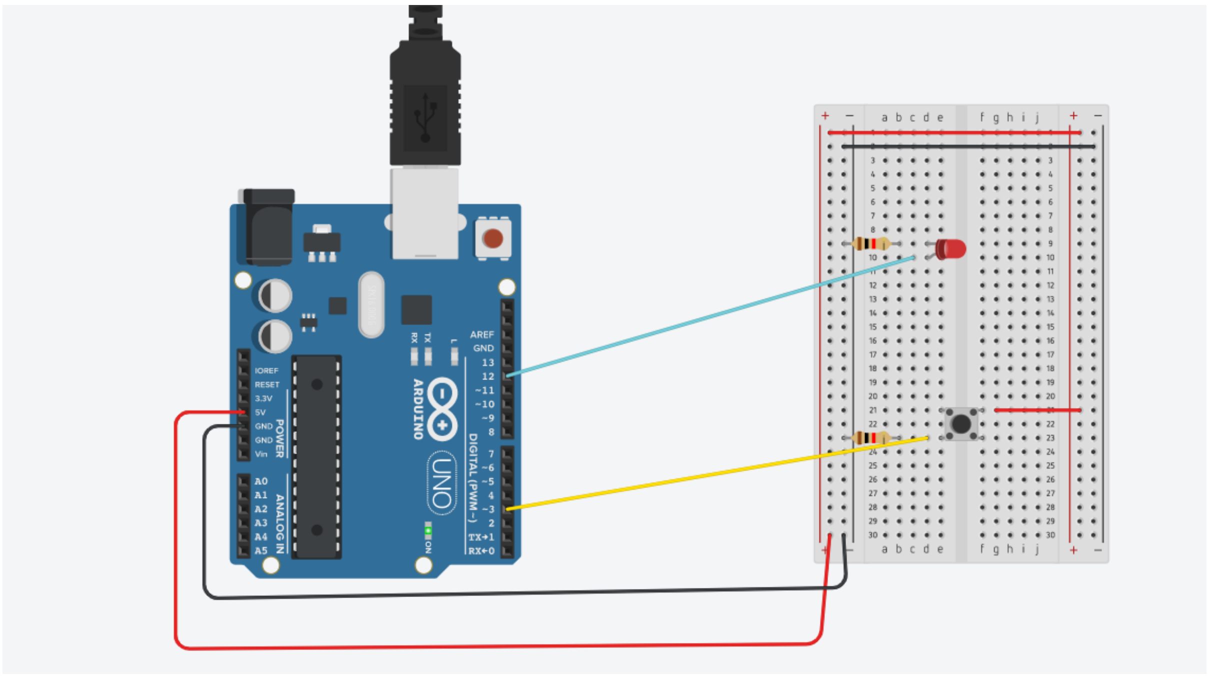
GREEN and YELLOW light on

(b) Develop a program to interface Input Switches and output LEDs with Arduino.

PROGRAM CODE:

```
int btn=0;
void setup()
{
  pinMode(3,INPUT);
  pinMode(12,OUTPUT);


}


void loop()
{
  btn=digitalRead(3);
  if(btn==HIGH)
  {digitalWrite(12,HIGH);
   delay(5000);


  }
  else
  {
    digitalWrite(12,LOW);


  }
}
```

OUTPUT:



PRACTICAL-6

AIM:

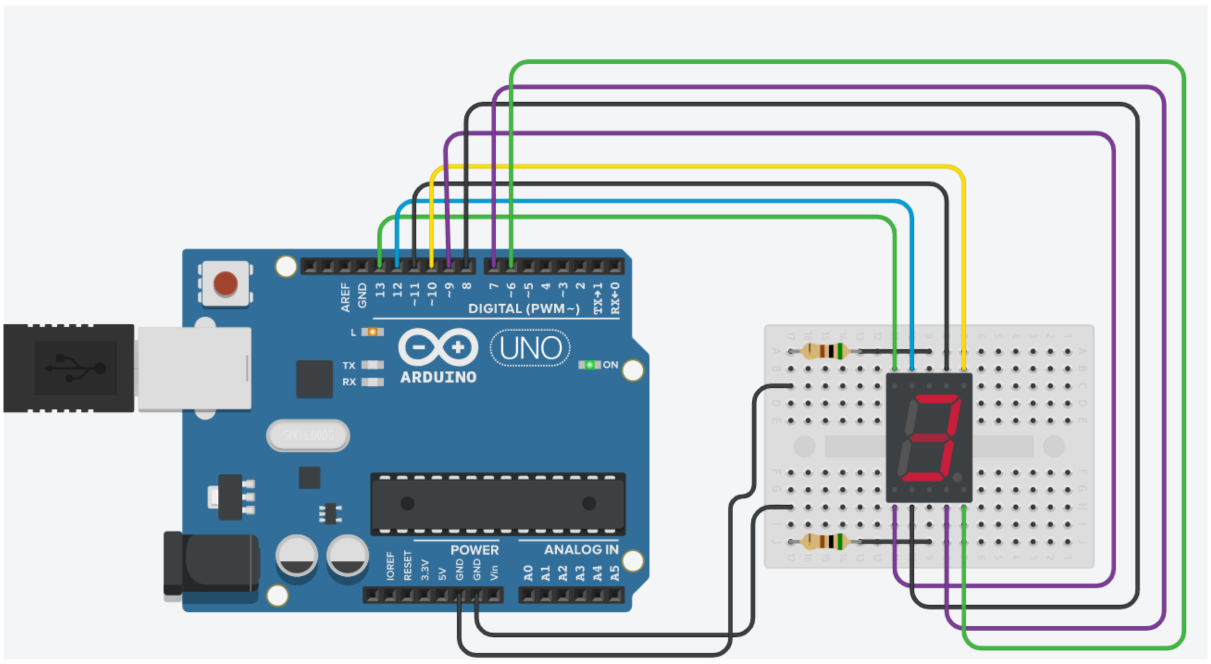
Interface 7 seg display with Arduino and Write a program to count and display 0 to 9 on it.

PROGRAM CODE:

```
unsigned const int A = 13;
unsigned const int B = 12;
unsigned const int C = 11;
unsigned const int D = 10;
unsigned const int E = 9;
unsigned const int F = 8;
unsigned const int G = 7;
unsigned const int H = 6;



void setup(void)
{
  pinMode(A, OUTPUT);
  pinMode(B, OUTPUT);
  pinMode(C, OUTPUT);
  pinMode(D, OUTPUT);
  pinMode(E, OUTPUT);
  pinMode(F, OUTPUT);
  pinMode(G, OUTPUT);
  pinMode(H, OUTPUT);
}



void zero(void) {
  digitalWrite(A, LOW);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, HIGH);
  digitalWrite(F, HIGH);
  digitalWrite(G, HIGH);
  digitalWrite(H, LOW);
}

void one(void) {
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
  digitalWrite(D, HIGH);
  digitalWrite(E, LOW);
  digitalWrite(F, LOW);
  digitalWrite(G, HIGH);
  digitalWrite(H, LOW);
}

void two(void) {
  digitalWrite(A, HIGH);
  digitalWrite(B, LOW);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, HIGH);
  digitalWrite(F, HIGH);
  digitalWrite(G, LOW);
  digitalWrite(H, LOW);
}

void three(void) {
  digitalWrite(A, HIGH);
  digitalWrite(B, LOW);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, LOW);
  digitalWrite(F, HIGH);
  digitalWrite(G, HIGH);
  digitalWrite(H, LOW);
}

void four(void) {
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, LOW);
  digitalWrite(D, HIGH);
  digitalWrite(E, LOW);
  digitalWrite(F, LOW);
  digitalWrite(G, HIGH);
  digitalWrite(H, LOW);
}

void five(void) {
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
  digitalWrite(F, HIGH);
  digitalWrite(G, HIGH);
  digitalWrite(H, LOW);
}

void six(void) {
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, LOW);
  digitalWrite(E, HIGH);
  digitalWrite(F, HIGH);
  digitalWrite(G, HIGH);
  digitalWrite(H, LOW);
}

void seven(void) {
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, LOW);
  digitalWrite(F, LOW);
  digitalWrite(G, HIGH);
  digitalWrite(H, LOW);
}

void eight(void) {
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, HIGH);
  digitalWrite(F, HIGH);
  digitalWrite(G, HIGH);
  digitalWrite(H, LOW);
}

void nine(void) {
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, LOW);
  digitalWrite(F, HIGH);
  digitalWrite(G, HIGH);
  digitalWrite(H, LOW);
}
```

```
void loop(void)
{
  zero();
  delay(1000);

  one();
  delay(1000);

  two();
  delay(1000);

  three();
  delay(1000);

  four();
  delay(1000);

  five();
  delay(1000);

  six();
  delay(1000);

  seven();
  delay(1000);

  eight();
  delay(1000);
  nine();
  delay(1000);
}
```

OUTPUT:



PRACTICAL-7

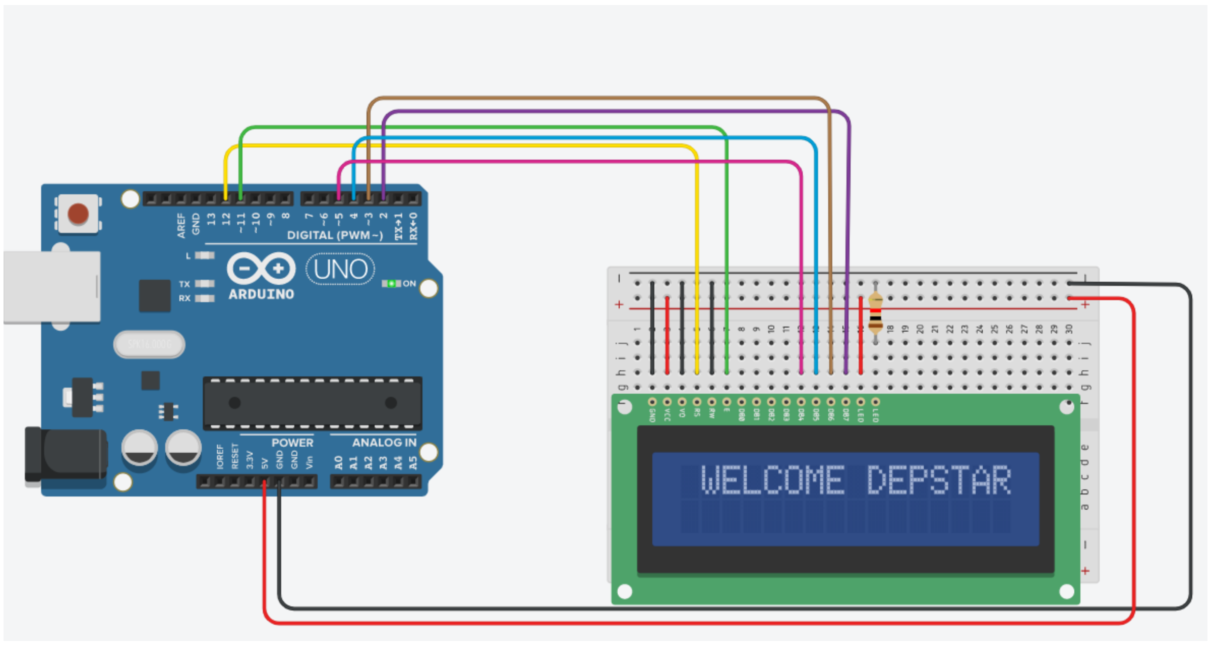
AIM:

Interface 16x2 LCD with Arduino. Write a Program to Display "WELCOME DEPSTAR".

PROGRAM CODE:

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(12,11,5,4,3,2);
void setup()
{
  lcd.begin(16,2);
}

void loop()
{
  lcd.setCursor(8,0);
  lcd.print("WELCOME DEPSTAR");
  lcd.scrollDisplayLeft();
  delay(1000);
}
```

OUTPTUT:



PRACTICAL-8

AIM:

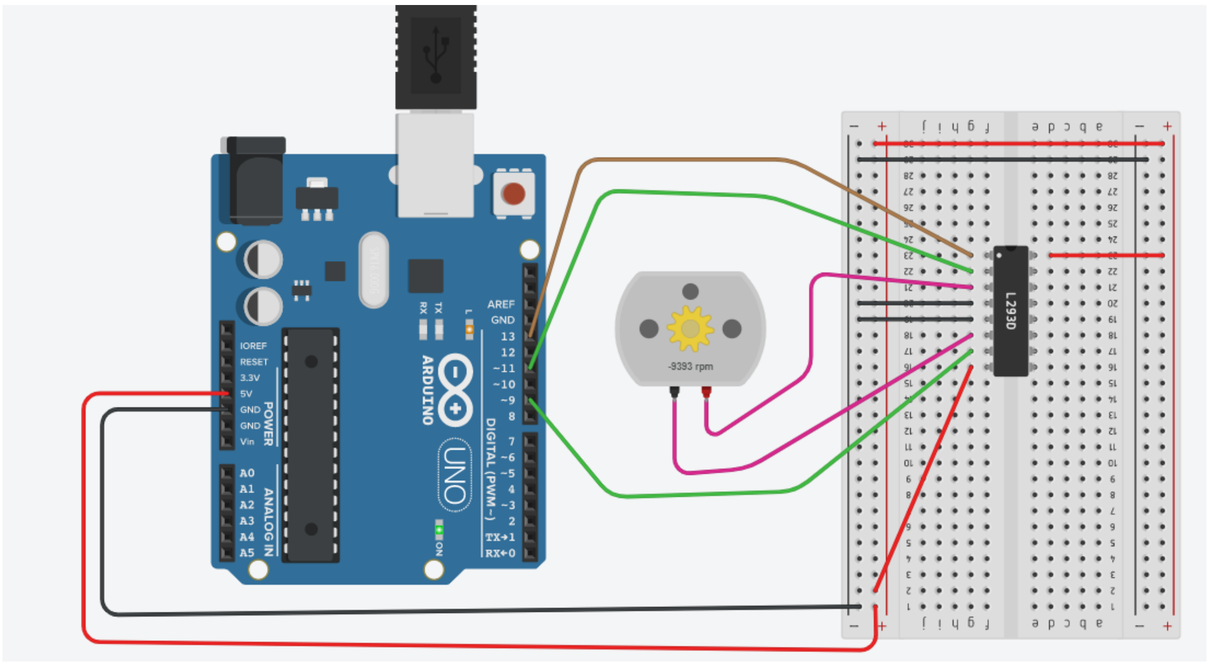
Interface Stepper motor and DC motor with Arduino and Write a Program to rotate motor into clockwise and anticlockwise.

PROGRAM CODE:

```
void setup()
{
  pinMode(13, OUTPUT);
  pinMode(11,OUTPUT);
  pinMode(9,OUTPUT);
  digitalWrite(13,HIGH);
}

void loop()
{
  digitalWrite(11,HIGH);
  digitalWrite(9,LOW);
  delay(3000);
  digitalWrite(11,LOW);
  digitalWrite(9,HIGH);
  delay(3000);
```

}

OUTPUT:



PRACTICAL-9

AIM:

Interface Different Sensors (Ultrasonic, PIR, Temperature) with Arduino and also write a Program for one application of each sensors.

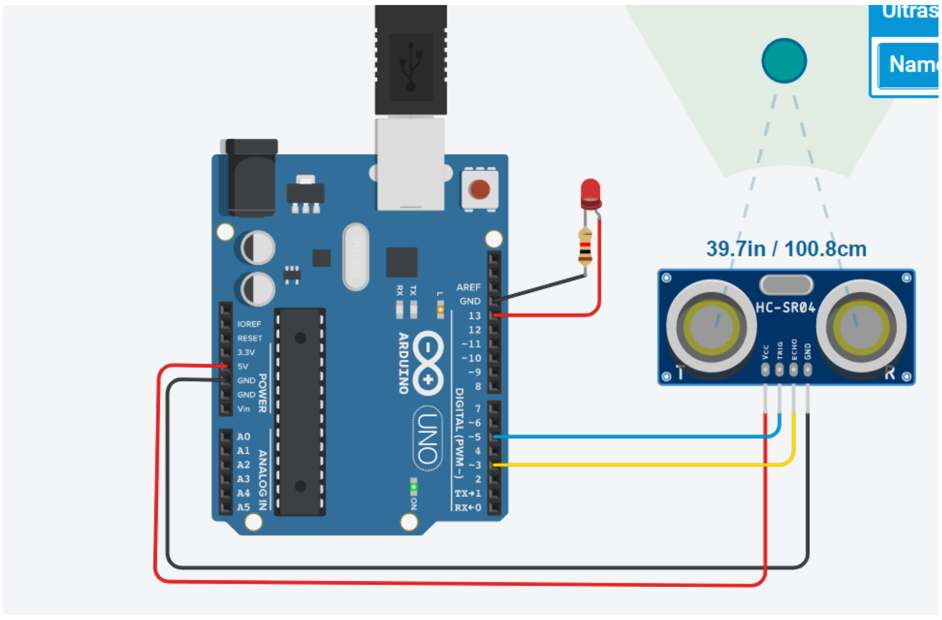
PROGRAM CODE:

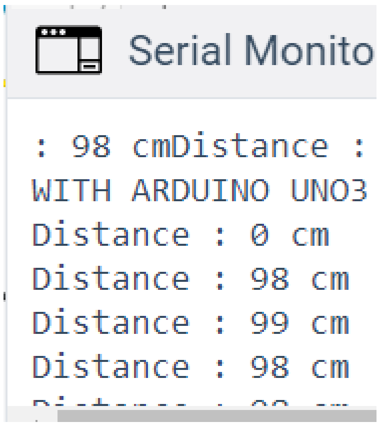
ULTRASONIC SENSOR:

```
#define echoPin 3
#define triggerPin 5
#define led 13

long duration;
int distance;
int val=0;

void setup()
{
  pinMode(led,OUTPUT);
  pinMode(echoPin,INPUT);
  pinMode(triggerPin,OUTPUT);
  Serial.begin(4800);
  Serial.println("ULTRASONIC SENSOR TEST");
  Serial.println("WITH ARDUINO UNO3");
}

void loop()
```

```
{
  digitalWrite(triggerPin,LOW);
  delayMicroseconds(2);
  digitalWrite(triggerPin,HIGH);
  delayMicroseconds(20);
  digitalWrite(triggerPin,LOW);
  val=digitalRead(3);
  duration=pulseIn(echoPin,HIGH);
  distance=(duration*0.034)/2;
  Serial.print("Distance : ");
  Serial.print(distance);
  Serial.print(" cm");

  if(distance<120)
  {
    digitalWrite(led,HIGH);
  }
  else
  {
    digitalWrite(led,LOW);
  }

}
```

OUTPUT:



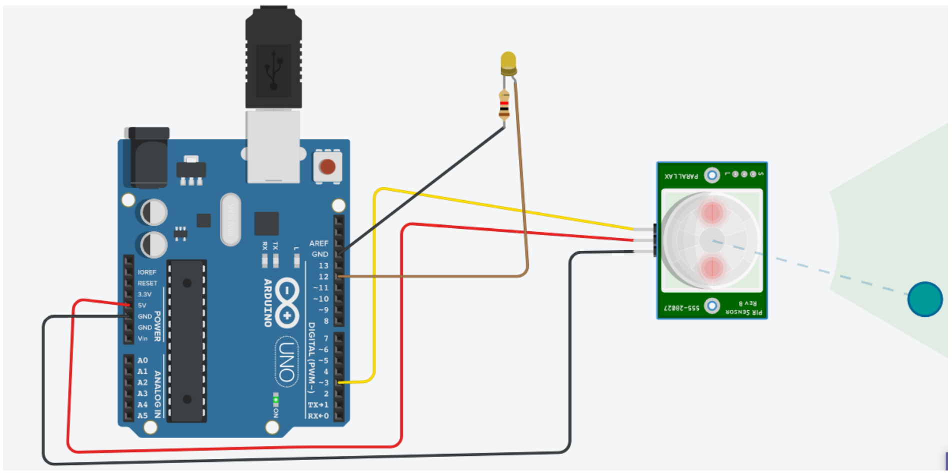


```
: 98 cmDistance :
WITH ARDUINO UNO3
Distance : 0 cm
Distance : 98 cm
Distance : 99 cm
Distance : 98 cm
```

PIR SENSOR:

```
int val=0;
void setup()
{
  pinMode(12, OUTPUT);
  pinMode(3,INPUT);
  Serial.begin(9600);
}

void loop()
{
  val=digitalRead(3);
  Serial.println(val);
  if(val==HIGH)
  {
    digitalWrite(12,HIGH);
    Serial.println("Sensor Activated");
  }
  else
  {digitalWrite(12,LOW);
   Serial.println("Sensor Deactivated");
  }
}
```

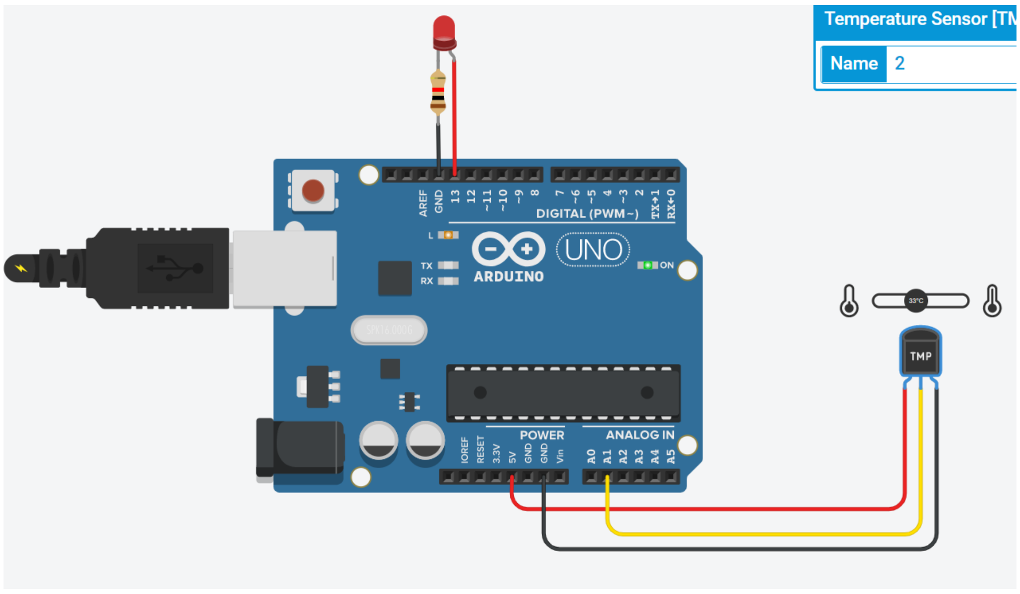
OUTPUT:

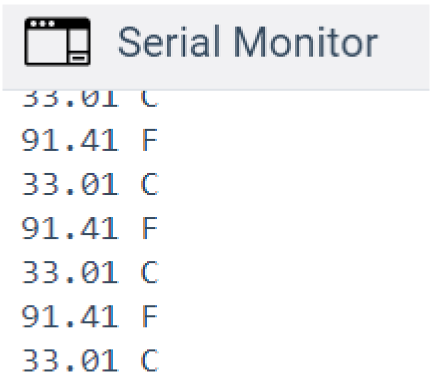


TEMPERATURE SENSOR:

```
float c,f;
void setup()
{
  pinMode(A1,INPUT);
  pinMode(13,OUTPUT);
  Serial.begin(9600);
}

void loop()
{

  c=analogRead(1);
  c=c*0.0048828125;
  c=(c-0.5)*100;
  Serial.print(c);
  Serial.println(" C");
  f=(9*c)/5+32;
  Serial.print(f);
  Serial.println(" F");
  if(c>=20)
  {
    digitalWrite(13,HIGH);
  }
  else
  {
    digitalWrite(13,LOW);
  }
```

OUTPUT:





```
33.01 C
91.41 F
33.01 C
91.41 F
33.01 C
91.41 F
33.01 C
```

PRACTICAL-10

AIM:

4 LEDs are Interfaced with Raspberry Pi. Develop a python script for Raspberry Pi to blink all LEDs with specific Time Interval.

PROGRAM CODE:

```
from goto import * import time import var import pio import resource import RPi.GPIO as GPIO

# Peripheral Configuration Code (do not edit)

#---CONFIG_BEGIN--- import cpu import FileStore import VFP ledpin1 = 18 ledpin2 = 19 ledpin3 = 20 ledpin4 = 12




GPIO.setmode(GPIO.BCM)

GPIO.setup(ledpin1,GPIO.OUT)

GPIO.setup(ledpin2,GPIO.OUT)

GPIO.setup(ledpin3,GPIO.OUT)

GPIO.setup(ledpin4,GPIO.OUT)

while True: try:

GPIO.output(ledpin1,GPIO.HIGH) time.sleep(1)

#GPIO.output(ledpin1,GPIO.LOW)

# time.sleep(1)

GPIO.output(ledpin2,GPIO.HIGH) time.sleep(1)

#GPIO.output(ledpin2,GPIO.LOW)

#time.sleep(1)

GPIO.output(ledpin3,GPIO.HIGH) time.sleep(1)

# GPIO.output(ledpin3,GPIO.LOW)

# time.sleep(1)

GPIO.output(ledpin4,GPIO.HIGH) time.sleep(1)
```

```
except:

print("Some Error!")



finally:

GPIO.cleanup()
```

PRACTICAL-11

AIM:

A 16 * 2 LCD is interfaced with Raspberry Pi. Develop a python script for Raspberry Pi to display the string on LCD. E.g., "Welcome to Charusat".

PROGRAM CODE:

```
import RPi.GPIO as GPIO import time


LCD_RS = 7

LCD_E = 8

LCD_D4 = 25

LCD_D5 = 24

LCD_D6 = 23

LCD_D7 = 18



# Define some device constants

LCD_WIDTH = 16 # Maximum characters per line

LCD_CHR = True
```

```
LCD_CMD = False



LCD_LINE_1 = 0x80 # LCD RAM address for the 1st line

LCD_LINE_2 = 0xC0 # LCD RAM address for the 2nd line

# Timing constants

E_PULSE = 0.0005

E_DELAY = 0.0005



def main():

# Main program block

GPIO.setwarnings(False)

        GPIO.setmode(GPIO.BCM)  # Use BCM GPIO numbers

GPIO.setup(LCD_E, GPIO.OUT) # E

GPIO.setup(LCD_RS, GPIO.OUT) # RS

GPIO.setup(LCD_D4, GPIO.OUT) # DB4

GPIO.setup(LCD_D5, GPIO.OUT) # DB5

GPIO.setup(LCD_D6, GPIO.OUT) # DB6

GPIO.setup(LCD_D7, GPIO.OUT) # DB7


# Initialise display lcd_init()


while True:



# Send some test lcd_string("Welcome to",LCD_LINE_1)
lcd_string("Charusat",LCD_LINE_2)
```

```
time.sleep(2)

def lcd_init(): # Initialise display

lcd_byte(0x33,LCD_CMD) # 110011 Initialise lcd_byte(0x32,LCD_CMD) # 110010
Initialise lcd_byte(0x06,LCD_CMD) # 000110 Cursor move direction
lcd_byte(0x0C,LCD_CMD) # 001100 Display On,Cursor Off, Blink Off

lcd_byte(0x28,LCD_CMD) # 101000 Data length, number of lines, font size
lcd_byte(0x01,LCD_CMD) # 000001 Clear display time.sleep(E_DELAY)



def lcd_byte(bits, mode):

# Send byte to data pins

# bits = data

# mode = True for character

        #       False for command


GPIO.output(LCD_RS, mode) # RS



# High bits

GPIO.output(LCD_D4, False)

GPIO.output(LCD_D5, False)

GPIO.output(LCD_D6, False)

GPIO.output(LCD_D7, False)

if bits&0x10==0x10:

GPIO.output(LCD_D4, True)

if bits&0x20==0x20:

GPIO.output(LCD_D5, True)

if bits&0x40==0x40:

GPIO.output(LCD_D6, True)

if bits&0x80==0x80:
 GPIO.output(LCD_D7, True)

# Toggle 'Enable' pin lcd_toggle_enable()

# Low bits
GPIO.output(LCD_D4, False)
GPIO.output(LCD_D5, False)
GPIO.output(LCD_D6, False)
GPIO.output(LCD_D7, False)
 if bits&0x01==0x01:
GPIO.output(LCD_D4, True)
if bits&0x02==0x02:
GPIO.output(LCD_D5, True)
if bits&0x04==0x04:
GPIO.output(LCD_D6, True)
if bits&0x08==0x08:
 GPIO.output(LCD_D7, True)

# Toggle 'Enable' pin lcd_toggle_enable()
 def lcd_toggle_enable(): # Toggle enable
time.sleep(E_DELAY)
GPIO.output(LCD_E, True)
 time.sleep(E_PULSE)
 GPIO.output(LCD_E, False)
time.sleep(E_DELAY)
 def lcd_string(message,line):
# Send string to display
```

```
message = message.ljust(LCD_WIDTH," ")
lcd_byte(line, LCD_CMD)
for i in range(LCD_WIDTH):
lcd_byte(ord(message[i]),LCD_CHR)


        if    name        == '  main ':
try:
main()
except KeyboardInterrupt:
pass
finally:
lcd_byte(0x01, LCD_CMD)
lcd_string("Goodbye!",LCD_LINE_1)
GPIO.cleanup()
```