کدها:

بخش اول و دوم

```
#include <Keypad.h>

#define LED1 10
#define LED2 9
#define LED3 8
#define LED4 7
#define LED5 6
#define LED6 5
#define LED7 4
#define LED8 3
#define LED9 2
const byte ROWS = 4; //four rows
const byte COLS = 4; //three columns
char keys[ROWS][COLS] = {
  {'7','8','9','%'},
  {'4','5','6','*'},
  {'1','2','3','-'},
  {'o','0','=','+'}
};
byte rowPins[ROWS] = {31, 33, 35, 37}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {29, 27, 25, 23}; //connect to the column pinouts of the keypad

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

void setup(){
  Serial.begin(9600);
  pinMode(LED1,OUTPUT);
  pinMode(LED2,OUTPUT);
  pinMode(LED3,OUTPUT);
  pinMode(LED4,OUTPUT);
  pinMode(LED5,OUTPUT);
  pinMode(LED6,OUTPUT);
  pinMode(LED7,OUTPUT);
  pinMode(LED8,OUTPUT);
  pinMode(LED9,OUTPUT);
}
void loop(){
  char key = keypad.getKey();
  int convertInt;

  if (key){
    /*Serial.println(key);*/
    convertInt = key - '0';
    digitalWrite(LED1,LOW);
    digitalWrite(LED2,LOW);
    digitalWrite(LED3,LOW);
    digitalWrite(LED4,LOW);
    digitalWrite(LED5,LOW);
    digitalWrite(LED6,LOW);
    digitalWrite(LED7,LOW);
    digitalWrite(LED8,LOW);
    digitalWrite(LED9,LOW);
    Serial.println(key);
    if (convertInt >0 and convertInt<10){
       for(int i=0;i< convertInt;i++){
          digitalWrite(LED1 - i,HIGH);
          delay(500);
       }

       }
    }

}
```

بخش سوم

```
#include <Keypad.h>

#define LED1 10
#define LED2 9
#define LED3 8
#define LED4 7
#define LED5 6
#define LED6 5
#define LED7 4
#define LED8 3
#define LED9 2
const byte ROWS = 4; //four rows
const byte COLS = 4; //three columns
char keys[ROWS][COLS] = {
  {'7','8','9','%'},
  {'4','5','6','*'},
  {'1','2','3','-'},
  {'o','0','=','+'}
};
byte rowPins[ROWS] = {31, 33, 35, 37}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {29, 27, 25, 23}; //connect to the column pinouts of the keypad

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

void setup(){
  Serial.begin(9600);
  pinMode(LED1,OUTPUT);
  pinMode(LED2,OUTPUT);
  pinMode(LED3,OUTPUT);
  pinMode(LED4,OUTPUT);
  pinMode(LED5,OUTPUT);
  pinMode(LED6,OUTPUT);
  pinMode(LED7,OUTPUT);
  pinMode(LED8,OUTPUT);
  pinMode(LED9,OUTPUT);
}

void loop(){
  int key ;
  int convertInt;

  if (Serial.available()>0){
     key = Serial.parseInt() - 0;
    if (key > -1 and key < 10){

    /*Serial.println(key);*/
    digitalWrite(LED1,LOW);
    digitalWrite(LED2,LOW);
    digitalWrite(LED3,LOW);
    digitalWrite(LED4,LOW);
    digitalWrite(LED5,LOW);
    digitalWrite(LED6,LOW);
    digitalWrite(LED7,LOW);
    digitalWrite(LED8,LOW);
    digitalWrite(LED9,LOW);
    for(int i=0;i< key;i++){
          digitalWrite(LED1 - i,HIGH);
          delay(500);
    }
  }else Serial.println("invalid number");
  }


}
```

انواع keypad و کارکرد آن‌ها:

1. Keypad ماتریسی یا دیجیتال: هنگامی که دکمه فشرده می‌شود سیم مربوط به ردیف آن کلید با سیم مربوط به ستون به هم متصل می‌شوند و هم ولتاژ خواهند شد و متوجه می‌شویم که کدام دکمه فشار داده شده است.
2. Keypad خازنی: هر دکمه یک خازن دارد که با فشار دادن دکمه خازن شارژ شده و می‌فهمیم که کدام دکمه فشار داده شده است.

Bounce چیست و راه‌های جلوگیری:

وقتی یک کلید فشرده شود ابتدا دو قطعه فلز به هم برخورد می‌کنند و باعث برقرار شدن جریان می‌شوند. سپس اگر چندین بار دکمه‌ها در مدت زمان کوتاه فشرده شوند (10 تا 100 بار در چند میکروثانیه)، باعث ایجاد نویز می‌شوند.

استفاده از یک خازن یا latch در بین راه پین‌ها تا ولتاژ ثابت شود و نوسانات را کاهش دهد. همچنین می‌توان به صورت نرم‌افزاری، مدت زمانی را تعیین کرد تا دستور مجددی نگیرد (مانند استفاده از کتابخانهٔ keypad).

تعریف توابع keypad.h:

keypad( ): این تابع با گرفتن پارامترهای آیکون‌های keypad، پین‌های ردیف‌ها و ستون‌ها و تعداد ردیف‌ها و ستون‌ها، یک شیء keypad می‌سازد.

getKey( ): کاراکتر فشرده شده را می‌دهد.

getKeys( ): دکمه‌هایی که همزمان فشرده شده‌اند را می‌دهد.

waitForKeys( ): تا گرفتن یک دکمه برنامه را متوقف می‌کند.

getState( ): حالت‌های هر کلید را نشان می‌دهد (hold, released, pressed, idle).

ketStateChanged( ): وضعیت هر کلید که تغییر کند را نشان می‌دهد با یک boolean.

نحوه و کاربردهای ارتباط سریال در آردوینو:

این سریال‌ها مجموعه‌ای از پین‌های مشخص هستند برای دریافت داده (Rx) و ارسال داده (Tx) و اطلاعاتی مثل buffer و Interrupt به صورت Sequential با یک کد binary می‌فرستد و بررسی می‌کند. کاربرد آن‌ها، ارتباط با دستگاه‌های خروجی دهنده مثل اسیلوسکوپ است.

تعریف مختصر و نحوهٔ کار با توابع ارتباط سریال:

begin(): سرعت انتقال برای تبادل داده‌ها را تعیین می‌کند و آغاز به کار می‌کند.

end(): تبادل اطلاعات را پایان می‌دهد.

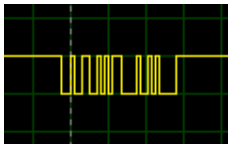
()find: جستجو در بافر و نمایش به صورت Boolean.
()parseInt: گرفتن مقدار عددی ورودی ترمینال.
()println: چاپ کردن و رفتن به خط بعدی ترمینال (مقدار را در Tx به صورت ASCII و یک newline).
()read: ورودی بعدی را از پورت Rx می‌خواند.
()readStringUntil: string موجود در Rx را تا هر زمان که ادامه داشته باشد را می‌خواند.
()write: داده‌ها را به صورت مجموعه‌ای از بایت‌ها در پورت Tx می‌نویسد.

در رابطه با نحوهٔ عملکرد سریال و در آردوینو و همچنین پروتکل‌های ارتباطی UART و USART توضیح مختصری ارائه دهید.

این دو پروتکل‌های ارتباطی سخت‌افزاری هستند که هدف آن‌ها دریافت و ارسال اطلاعات و داده‌ها با پورت‌های Tx و Rx به صورت سیگنال است.

USART داده‌ها را به شکل Synchronize یعنی هماهنگ و همزمان ارسال و دریافت می‌کند. ولی UART به صورت Asynchronize یعنی ناهماهنگ و غیرهمزمان است چون USART همزمان و با یک کلاک است و سرعت بالاتری نسبت به UART دارد.

سوال 2 گزارشکار:

هنگام فشردن کلید، A موج  در اسیلوسکوپ دیده می‌شود.