https://mrarroyoworldwonders.weebly.com/uploads/1/2/0/3/120363503/reading_-_chapter_22.pdf

```
'{$STAMP BS2}
'{$PBASIC 2.5}

OUTPUT 9
initLED VAR OUT9
OUTPUT 6
timeToDischarge VAR word
counter VAR word
TURNON CON 1
TURNOFF CON 0
x VAR byte
'blink the LED
initLED = TURNON
PAUSE 500
initLED = TURNOFF
PAUSE 500
  FREQOUT 10, 1000, 2000
  FREQOUT 10, 600, 2000

MAIN:
DO
  DO
    PULSOUT 12, 830
    PULSOUT 13, 650
    HIGH 6
    PAUSE 3
    RCTIME 6,1, timeToDischarge
    DEBUG ? timeToDischarge
  LOOP WHILE timeToDischarge < 7
  FOR x = 1 to 8
    PULSOUT 12, 800
    PULSOUT 13, 800
    HIGH 6
    PAUSE 20
    RCTIME 6, 1, timeToDischarge
    DEBUG ? timeToDischarge
    if timeToDischarge < 7 then
      EXIT
    ENDIF
  NEXT
  if timeToDischarge > 6 then
    FOR x = 1 to 18
      PULSOUT 12, 700
      PULSOUT 13, 700
      HIGH 6
      PAUSE 20
      RCTIME 6, 1, timeToDischarge
      if timeToDIscharge < 6 then
        EXIT
      ENDIF
    NEXT
 ENDIF
LOOP
```

Subroutines:

```
moveForward:
FOR x = 1 to 10
  PULSOUT 12, 830
  PULSOUT 13, 650
NEXT


'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
TURNON CON 1
TURNOFF CON 0
PRESSED CON 1
NOTPRESSED CON 0
LIFTED CON 1
DETECTED CON 1
THREESECONDS CON 256
x VAR word
pulseCount VAR word
EE_address VAR byte
instruction VAR byte

'-----------initialization---------
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
```

```
FREQOUT 10, 1000, 1500

'------------main------------------'
INPUT 3
pushButton VAR IN3
OUTPUT 9
LED VAR OUT9

counter VAR word
timeToDischarge var Word
OUTPUT 6
DO
  PULSOUT 12, 850
  PULSOUT 13, 650
  HIGH 6
  PAUSE 3
  RCTIME 6, 1, timeToDischarge
  DEBUG ? timeToDischarge
LOOP WHILE timeToDischarge < 24
GOSUB turnRight
DO
  PULSOUT 12, 850
  PULSOUT 13, 650
  HIGH 6
  PAUSE 3
  RCTIME 6, 1, timeToDischarge
  DEBUG ? timeToDischarge
LOOP WHILE timeToDischarge < 17
GOSUB moveAdjust1
GOSUB turnLeft
DO
  PULSOUT 12, 850
  PULSOUT 13, 650
  HIGH 6
  PAUSE 3
  RCTIME 6, 1, timeToDischarge
  DEBUG ? timeToDischarge
LOOP WHILE timeToDischarge > 2
GOSUB moveAdjust2
GOSUB turnRight
DO
  PULSOUT 12, 850
  PULSOUT 13, 650
  HIGH 6
  PAUSE 3
  RCTIME 6, 1, timeToDischarge
  DEBUG ? timeToDischarge
LOOP WHILE timeToDischarge < 12
GOSUB stopMove
END


'--------------subroutines--------------'

counting:
  DO WHILE pushButton = PRESSED
    DEBUG "pressed", CR
    counter = counter + 1
  LOOP
RETURN

overThreshold:
  DEBUG ? counter
  DEBUG "pushbutton held more than 3 seconds, entering low power mode"
RETURN

underThreshold:
  DEBUG ? counter
  DEBUG "pushbutton held less than 3 seconds, entering low power mode"
RETURN

moveForward:
  FOR x = 1 to 100:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
moveAdjust2:
  FOR x = 1 to 40:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
moveAdjust1:
  FOR x = 1 to 5:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
```

```
moveBackward:
  FOR x = 1 to 95:
    PULSOUT 12, 650
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN

turnLeft:
  FOR x = 1 to 23:
    PULSOUT 12, 650
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN

turnRight:
  FOR x = 1 to 21:
    PULSOUT 12, 850
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN

stopMove:
FOR x = 1 to 20
  PULSOUT 12, 750
  PULSOUT 13, 750
  PAUSE 20
NEXT
RETURN
```

```
D7S:
 FREQOUT 10, 200, 2489
RETURN
E7:
 FREQOUT 10, 200, 2637
RETURN
B:
 FREQOUT 10, 200, 1975
RETURN
D7:
 FREQOUT 10, 200, 2349
RETURN
C7:
 FREQOUT 10, 200, 2043
RETURN
A6:
 FREQOUT 10, 200, 1760
RETURN
C6:
 FREQOUT 10, 200, 1046
RETURN
E6:
 FREQOUT 10, 200, 1318
RETURN
G6S:
 FREQOUT 10, 200, 1661
RETURN
BSeven:
 FREQOUT 10, 200, 3591
RETURN
F7S:
 FREQOUT 10, 200, 2960
```

```
RETURN
A7:
  FREQOUT 10, 200, 3520
RETURN
A7S:
  FREQOUT 10, 200, 3729
RETURN
playSong:
  GOSUB E7
  GOSUB D7S
  GOSUB E7
  GOSUB D7S
  GOSUB E7
  GOSUB B
  GOSUB D7
  GOSUB C7
  GOSUB A6
  PAUSE 600
  GOSUB A6
  PAUSE 600
  GOSUB A6
  PAUSE 600
  GOSUB A6
  GOSUB C6
  GOSUB E6
  GOSUB A6
  GOSUB B
  PAUSE 600
  GOSUB B
  PAUSE 600
  GOSUB B
  PAUSE 600
  GOSUB B
  GOSUB E6
  GOSUB G6S
  GOSUB B
  GOSUB C7
  PAUSE 600
  GOSUB C7
  PAUSE 600
  GOSUB E7
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
timeToDischarge var Word
OUTPUT 6
DO
  HIGH 6
  PAUSE 3
  RCTIME 6, 1, timeToDischarge
  DEBUG ? timeToDischarge
LOOP
```

```
INPUT 3
pushButton VAR IN3
OUTPUT 9
LED VAR OUT9

counter VAR word
```

```
main:
  DO WHILE pushButton = NOTPRESSED
    DEBUG "not pressed", CR
  LOOP
  GOSUB counting
  IF counter > THREESECONDS THEN
    GOSUB overThreshold
  ELSE
    GOSUB underThreshold
  ENDIF
END

'--------------subroutines--------------'

counting:
  DO WHILE pushButton = PRESSED
    DEBUG "pressed", CR
    counter = counter + 1
  LOOP
RETURN

overThreshold:
  DEBUG ? counter
  DEBUG "pushbutton held more than 3 seconds, entering low power
mode"
RETURN

underThreshold:
  DEBUG ? counter
  DEBUG "pushbutton held less than 3 seconds, entering low power
mode"
```

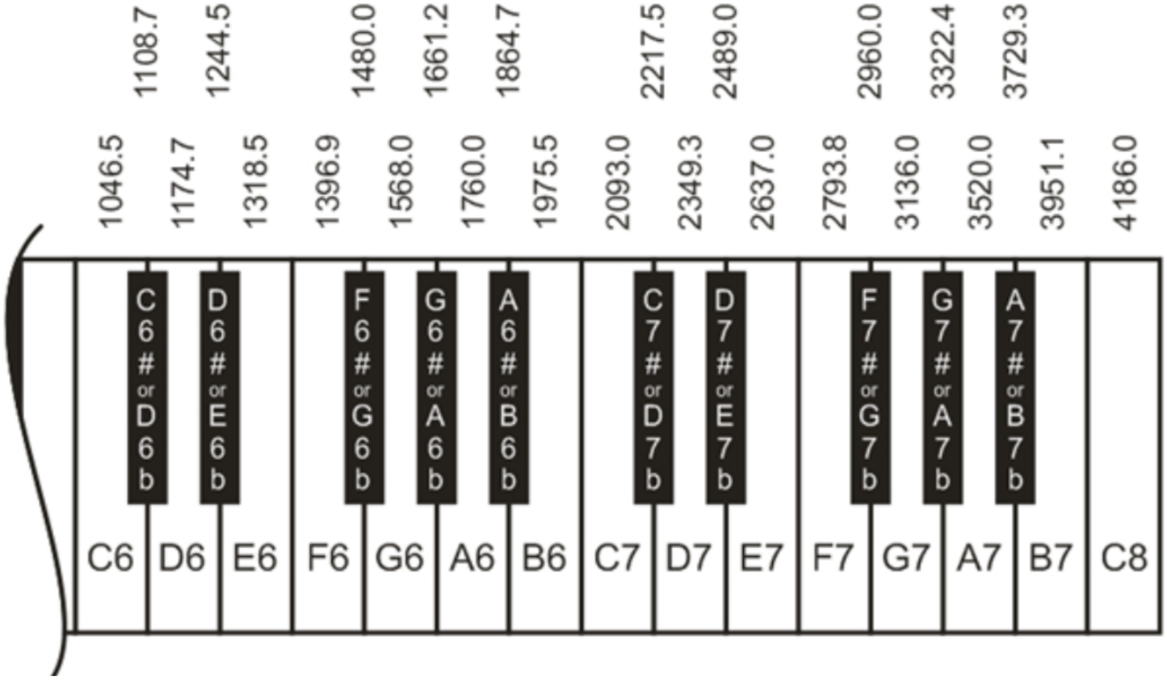
```
RETURN

' {$STAMP BS2}
' {$PBASIC 2.5}
' -----[ Code Title
]-----------------------------------------------------------------------
' Timer Lab --- Eric Pang Period 1 Reid
' -----[ Variables/Constants]--------------------------------------------------
OUTPUT 11
  initLed VAR OUT11 'Port for initialization LED
INPUT 3
  pushButton var IN3
TURNON CON 1 'Turn LED's on or off
TURNOFF CON 0
PRESSED CON 1 'PushButton action
NOTPRESSED CON 0
counter var word 'Variable tracking how long the push button has been
pressed
' -----[ Main Routine ]-------------------------------------------------------------
main:
GOSUB init 'Initialization
  DO
   DEBUG "not pressed", CR 'Say the button is not pressed in the
debug window
   LOOP WHILE (pushButton = NOTPRESSED) 'Loop saying "not
pressed" while the pushButton isn't pressed
   GOSUB counting 'Enter subroutine once the button is not pressed
  IF counter > 3992 THEN 'If the time pressed is more than 3 seconds
then...
     GOSUB overThreshold
  ELSE
     GOSUB underThreshold
  ENDIF
' -----[ Sub Routines ]----------------------------------------------------------------
counting:
DO
  counter = counter + 1
  loop while (pushButton = PRESSED) 'Begin counting once the
pushbutton is pressed
RETURN

underThreshold:
  DEBUG ? counter 'Say "counter = # of how long pressed"
  DEBUG "pushbutton held for less than 3 seconds", CR
  DEBUG "now entering low power mode"
  END 'Low power mode
RETURN

overThreshold:
  DEBUG ? counter 'Say "counter = #"
  DEBUG "pushbutton held for 3 seconds or more", CR
  DEBUG "now entering low power mode"
  END 'Low power mode
RETURN

init:
  initLED = TURNON 'Blinks the LED on
    PAUSE 500
  initLed = TURNOFF'Turns off the LED
    PAUSE 500
  FREQOUT 10, 1000, 2000
    INPUT 4
Return
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
TURNON CON 1
TURNOFF CON 0
PRESSED CON 0
LIFTED CON 1
DETECTED CON 1
x VAR word
pulseCount VAR word
EE_address VAR byte
instruction VAR byte

'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
'FREQOUT 10, 1000, 1500

INPUT 3
photoResistor VAR IN3
OUTPUT 9
LED VAR OUT9

counter VAR byte
counter2 VAR byte
'GOSUB playSong
```

```
'END
DO WHILE photoResistor = 0
  PULSOUT 12, 850
  PULSOUT 13, 650
  PAUSE 20
  counter = counter + 1
LOOP
GOSUB stopMove
GOSUB turnLeft
DO WHILE photoResistor = 0
  PULSOUT 12, 850
  PULSOUT 13, 650
  PAUSE 20
  counter2 = counter2 + 1
LOOP
GOSUB playSong
FOR x = 1 to counter2
  PULSOUT 12, 650
  PULSOUT 13, 850
  PAUSE 20
NEXT
GOSUB stopMove
GOSUB turnLeft
FOR x = 1 to counter
  PULSOUT 12, 850
  PULSOUT 13, 652
  PAUSE 20
NEXT
END
checkLight:

moveForward:
  FOR x = 1 to 100:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
moveBackward:
  FOR x = 1 to 95:
    PULSOUT 12, 650
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN
turnLeft:
  FOR x = 1 to 23:
    PULSOUT 12, 650
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
turnRight:
  FOR x = 1 to 28:
    PULSOUT 12, 850
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN
stopMove:
FOR x = 1 to 20
  PULSOUT 12, 750
  PULSOUT 13, 750
  PAUSE 20
```

```
NEXT
RETURN
D7S:
  FREQOUT 10, 200, 2489
RETURN
E7:
  FREQOUT 10, 200, 2637
RETURN
B:
  FREQOUT 10, 200, 1975
RETURN
D7:
  FREQOUT 10, 200, 2349
RETURN
C7:
  FREQOUT 10, 200, 2043
RETURN
A6:
  FREQOUT 10, 200, 1760
RETURN
C6:
  FREQOUT 10, 200, 1046
RETURN
E6:
  FREQOUT 10, 200, 1318
RETURN
G6S:
  FREQOUT 10, 200, 1661
RETURN
BSeven:
  FREQOUT 10, 200, 3591
RETURN
```

```
F7S:
  FREQOUT 10, 200, 2960
RETURN
A7:
  FREQOUT 10, 200, 3520
RETURN
A7S:
  FREQOUT 10, 200, 3729
RETURN
playSong:
  GOSUB E7
  GOSUB D7S
  GOSUB E7
  GOSUB D7S
  GOSUB E7
  GOSUB B
  GOSUB D7
  GOSUB C7
  GOSUB A6
  PAUSE 600
  GOSUB A6
  PAUSE 600
  GOSUB A6
  PAUSE 600
  GOSUB A6
  GOSUB C6
  GOSUB E6
  GOSUB A6
  GOSUB B
  PAUSE 600
  GOSUB B
  PAUSE 600
  GOSUB B
  PAUSE 600
  GOSUB B
  GOSUB E6
  GOSUB G6S
  GOSUB B
  GOSUB C7
  PAUSE 600
  GOSUB C7
  PAUSE 600
  GOSUB E7
  GOSUB D7S
  GOSUB E7
  GOSUB D7S
  GOSUB E7
  GOSUB B
  GOSUB D7
  GOSUB C7
  FREQOUT 10, 2000, 1760
RETURN
```



Every single note
C6 CON 1046
C6sharp CON 1108
D6 CON 1174
D6sharp CON 1244
E6 CON 1318
F6 CON 1396
F6sharp CON 1480
G6 CON 1568
G6sharp CON 1661
A6 CON 1760
A6sharp CON 1864
BE6 CON 1975
C7 CON 2093
C7sharp CON 2217
D7 CON 2349
D7sharp 2489
E7 CON 2637

```
F7 CON 2793
F7sharp CON 2960
G7 CON 3136
G7sharp CON 3322
A7 CON 3520
A7sharp CON 3729
BE7 CON 3951
C8 CON 4186
```

(BE is the same as B)

```
' {$STAMP BS2}
' {$PBASIC 2.5}
' -----[ Title ]-----------------------------------------------------------------

' What's a Microcontroller - YANKEE DOODLE


' -----[ Variables/Constants/Pins
]--------------------------------------------------


FreqDetectable  CON 3000

C6          CON 1047              ' Piano notes
D6          CON 1175
E6          CON 1319
F6          CON 1397
G6          CON 1568
A6          CON 1760
Be6         CON 1976

C7          CON 2093
D7          CON 2349
```

```
E7        CON  2637
F7             CON  2793
G7        CON  3136
A7        CON  3520
Be7            CON  3951
C8        CON  4186           ' end of piano notes

Piezospeaker   PIN  4         ' Speaker


' -----[ Main Routine ]-------------------------------------------------------------

FREQOUT 10, 500,C7
PAUSE 50
FREQOUT 10, 500,C7
PAUSE 50
FREQOUT 10, 500,D7
PAUSE 50
FREQOUT 10, 500,E7
PAUSE 50
FREQOUT 10, 500,C7
PAUSE 50
FREQOUT 10, 500,E7
PAUSE 50
FREQOUT 10, 1000,D7
PAUSE 50
FREQOUT 10, 500,C7
PAUSE 50
FREQOUT 10, 500,C7
PAUSE 50
FREQOUT 10, 500,D7
```

```
PAUSE 50
FREQOUT 10, 500,E7
PAUSE 50
FREQOUT 10, 500,C7
PAUSE 50
FREQOUT 10, 1000,BE6
PAUSE 50
FREQOUT 10, 500,C7
PAUSE 50
FREQOUT 10, 500,C7
PAUSE 50
FREQOUT 10, 500,D7
PAUSE 50
FREQOUT 10, 500,E7
PAUSE 50
FREQOUT 10, 500,F7
PAUSE 50
FREQOUT 10, 500,E7
PAUSE 50
FREQOUT 10, 500,D7
PAUSE 50
FREQOUT 10, 500,C7
PAUSE 50
FREQOUT 10, 500,BE6
PAUSE 50
FREQOUT 10, 500,G6
PAUSE 50
FREQOUT 10, 500,A6
PAUSE 50
FREQOUT 10, 500,BE6
PAUSE 50
FREQOUT 10, 1000,C7
```

```
PAUSE 50
FREQOUT 10, 1000,C7
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
TURNON CON 1
TURNOFF CON 0
PRESSED CON 0
LIFTED CON 1
DETECTED CON 1
x VAR word

'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
INPUT 3
photoResistor VAR IN3
OUTPUT 9
LED VAR OUT9

DO WHILE photoResistor = 0
  PULSOUT 12, 850
  PULSOUT 13, 650
  PAUSE 20
LOOP
GOSUB turnLeft
DO WHILE photoResistor = 0
  PULSOUT 12, 850
  PULSOUT 13, 650
  PAUSE 20
LOOP
END
```

```
moveForward:
  FOR x = 1 to 100:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
moveBackward:
  FOR x = 1 to 95:
    PULSOUT 12, 650
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN
turnLeft:
  FOR x = 1 to 22:
    PULSOUT 12, 650
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
turnRight:
  FOR x = 1 to 28:
    PULSOUT 12, 850
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN
stopMove:
  for x = 1 to 100:
    PULSOUT 12, 850 - x
    PULSOUT 13, 650 + x
    PAUSE 20
  NEXT
RETURN
Circle:
  for x = 1 to 480:
    PULSOUT 12, 850
    PULSOUT 13, 720
    PAUSE 20
  NEXT
'Sine Code
moveForwardRight:
  for x = 1 to 95:
    PULSOUT 12, 850
    PULSOUT 13, 720
    PAUSE 20
  NEXT
RETURN

moveForwardRight2:
  for x = 1 to 140:
    PULSOUT 12, 850
    PULSOUT 13, 725
    PAUSE 20
  NEXT
RETURN

moveForwardLeft:
  for x = 1 to 160:
    PULSOUT 12, 770
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN

moveForwardLeft2:
  for x = 1 to 150:
    PULSOUT 12, 770
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
```

```
{$STAMP BS2}
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
```

```
TURNON CON 1
TURNOFF CON 0
PRESSED CON 0
LIFTED CON 1
DETECTED CON 1
x VAR word

'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
INPUT 4
photoResistor VAR IN4
OUTPUT 6
LED VAR OUT6

DO
  IF photoResistor = DETECTED then
    LED = turnOn
    PAUSE 500
    LED = turnOff
    GOSUB moveForward
    PAUSE 400
    GOSUB moveBackward
  ENDIF
LOOP
END

moveForward:
  FOR x = 1 to 100:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
moveBackward:
  FOR x = 1 to 95:
    PULSOUT 12, 650
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN
turnLeft:
  FOR x = 1 to 22:
    PULSOUT 12, 650
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
turnRight:
  FOR x = 1 to 28:
    PULSOUT 12, 850
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN
stopMove:
  for x = 1 to 100:
    PULSOUT 12, 850 - x
    PULSOUT 13, 650 + x
    PAUSE 20
  NEXT
RETURN
Circle:
  for x = 1 to 480:
    PULSOUT 12, 850
    PULSOUT 13, 720
    PAUSE 20
  NEXT
'Sine Code
moveForwardRight:
  for x = 1 to 95:
    PULSOUT 12, 850
    PULSOUT 13, 720
    PAUSE 20
  NEXT
RETURN

moveForwardRight2:
  for x = 1 to 140:
    PULSOUT 12, 850
    PULSOUT 13, 725
    PAUSE 20
  NEXT
RETURN

moveForwardLeft:
  for x = 1 to 160:
    PULSOUT 12, 770
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN

moveForwardLeft2:
  for x = 1 to 150:
    PULSOUT 12, 770
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants and variables
TURNON CON 1
TURNOFF CON 0
x VAR byte
pulsecount var word
EE_address var byte
instruction var byte
'--------------------------------------------------
data "ifsrfsrfsr"  'EEPROM cmds

'-----------------------subroutines-go-here--------------------------------------------------------
main:
  read EE_address, instruction
  EE_address = EE_address + 1
    If instruction = "i" then reset
    If instruction = "s" then stopmove
    If instruction = "f" then moveForward
    If instruction = "l" then turnLeft
    If instruction = "r" then turnRight
stop
```

```
reset: 'initialization
  OUTPUT 11
  initLed VAR OUT11
  initLED = TURNON 'FLASH
  PAUSE 500
  initLed = TURNOFF
  PAUSE 500
  FREQOUT 10, 1000, 2000
  goto main

moveForward:
    FOR pulseCount = 0 to 200 step 2
      PULSOUT 12, 750 + pulseCount
      PULSOUT 13, 750 - pulseCount
      PAUSE 20
      next
  goto main

stopmove:
  FOR pulseCount = 0 to 100 step 2
      PULSOUT 12, 850 - pulseCount
      PULSOUT 13, 650 + pulseCount
      PAUSE 20
    next
  goto main

 moveReverse:
  FOR x = 1 to 101
      PULSOUT 12, 650 'left
      PULSOUT 13, 850 'right
      PAUSE 20
    next
  goto main

turnLeft:
  FOR x = 1 to 58
      PULSOUT 12, 650 'left
      PULSOUT 13, 750 'right
      PAUSE 20
    next
  goto main

turnRight:
  FOR x = 1 to 58
      PULSOUT 12, 750 'left
      PULSOUT 13, 850 'right
      PAUSE 20
    next
  goto main



'{$STAMP BS2}
'{$PBASIC 2.5}
'constants and variables
TURNON CON 1
TURNOFF CON 0
x VAR byte
pulsecount var word
EE_address var byte
instruction var byte
'--------------------------------------------------

'-------------------------------------
data "ifsrfsrfsrfsr"

'-----------------------subroutines here--------------------------------------------------------
main:
  read EE_address, instruction
  EE_address = EE_address + 1

  If instruction = "i" then reset
  If instruction = "s" then stopmove
  If instruction = "f" then moveForward
  If instruction = "l" then turnLeft
  If instruction = "r" then turnRight
stop

reset: 'initialization
  OUTPUT 11
  initLed VAR OUT11
  'FLASH
  initLED = TURNON
  PAUSE 500
  initLed = TURNOFF
  PAUSE 500
  FREQOUT 10, 1000, 2000
  goto main

moveForward:
    FOR pulseCount = 0 to 200 step 2
      PULSOUT 12, 750 + pulseCount
      PULSOUT 13, 750 - pulseCount
      PAUSE 20
      next
  goto main

stopmove:
  FOR pulseCount = 0 to 100 step 2
      PULSOUT 12, 850 - pulseCount
      PULSOUT 13, 650 + pulseCount
      PAUSE 20
    next
  goto main

 moveReverse:
  FOR x = 1 to 101
      PULSOUT 12, 650 'left
      PULSOUT 13, 850 'right
      PAUSE 20
    next
  goto main

turnLeft:
  FOR x = 1 to 43
      PULSOUT 12, 650 'left
      PULSOUT 13, 750 'right
      PAUSE 20
    next
  goto main

turnRight:
  FOR x = 1 to 43
      PULSOUT 12, 750 'left
      PULSOUT 13, 850 'right
      PAUSE 20
    next
  goto main
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
TURNON CON 1
TURNOFF CON 0
PRESSED CON 0
LIFTED CON 1
x VAR word
pulse_count VAR word
EE_address VAR byte
instruction VAR byte

'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000

Main:
DATA "SFSLSFSRSFSB"
read EE_address, instruction
EE_address = EE_address + 1
if instruction = "F" then moveForward
if instruction = "B" then moveBackward
if instruction = "R" then turnRight
if instruction = "L" then turnLeft
if instruction = "S" then stopMove
STOP

moveForward:
  FOR x = 1 to 100:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
GOTO Main
moveBackward:
  FOR x = 1 to 100:
    PULSOUT 12, 650
    PULSOUT 13, 850
    PAUSE 20
  NEXT
GOTO Main
turnLeft:
  FOR x = 1 to 23
    PULSOUT 12, 650
    PULSOUT 13, 650
    PAUSE 20
  NEXT
GOTO Main
turnRight:
  FOR x = 1 to 22
    PULSOUT 12, 850
    PULSOUT 13, 850
    PAUSE 20
  NEXT
GOTO Main
stopMove:
  PULSOUT 12, 750
  PULSOUT 13, 750
  PAUSE 1000
GOTO Main
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants and variables
TURNON CON 1
```

```
TURNOFF CON 0
x VAR byte
pulse_count var word
EE_address var byte
'----------------------------------------------------

'--------------------------------------
data "isfslsfsrsfsr"

'------------------------subroutines here------------------------------------------------------------
main:
  read EE_address, instruction
  EE_address = EE_address + 1

  If instruction = "i" then reset
  If instruction = "s" then stopmove
  If instruction = "f" then moveForward
  If instruction = "l" then turnLeft
  If instruction = "r" then turnRight
stop

reset: 'initialization
  OUTPUT 11
  initLed VAR OUT11
  'FLASH
  initLED = TURNON
  PAUSE 500
  initLed = TURNOFF
  PAUSE 500
  FREQOUT 10, 1000, 2000
  goto main

stopmove:
      PULSOUT 12, 750
      PULSOUT 13, 750
      PAUSE 1000
  goto main

moveForward:
  FOR x = 1 to 101
      PULSOUT 12, 850 'left
      PULSOUT 13, 650 'right
      PAUSE 20
    next
  goto main

 moveReverse:
  FOR x = 1 to 101
      PULSOUT 12, 650 'left
      PULSOUT 13, 850 'right
      PAUSE 20
    next
  goto main

turnLeft:
  FOR x = 1 to 43
      PULSOUT 12, 650 'left
      PULSOUT 13, 750 'right
      PAUSE 20
    next
  goto main

turnRight:
  FOR x = 1 to 43
      PULSOUT 12, 750 'left
      PULSOUT 13, 850 'right
      PAUSE 20
    next
  goto main
```

MAKE IT MOVE:

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants and variables
TURNON CON 1
TURNOFF CON 0
x VAR byte
'-------------------------------------------------------------------------------------
GOSUB reset
GOSUB stopmove
GOSUB moveForward
GOSUB stopmove
GOSUB turnLeft
GOSUB stopmove
GOSUB moveForward
GOSUB stopmove
GOSUB turnRight
gosub stopmove
gosub moveForward
gosub stopmove
gosub moveReverse
END 'enter low power mode
'----------------------subroutines here-------------------------------------------------------------
'initialization
reset:
  OUTPUT 11
  initLed VAR OUT11
  'FLASH
  initLED = TURNON
  PAUSE 500
  initLed = TURNOFF
  PAUSE 500
  FREQOUT 10, 1000, 2000
  RETURN

stopmove:
      PULSOUT 12, 750
      PULSOUT 13, 750
      PAUSE 1000
  return

moveForward:
  FOR x = 1 to 101
      PULSOUT 12, 850 'left
      PULSOUT 13, 650 'right
      PAUSE 20
    next
  return

 moveReverse:
  FOR x = 1 to 101
      PULSOUT 12, 650 'left
      PULSOUT 13, 850 'right
      PAUSE 20
    next
  return
```

```
turnLeft:
  FOR x = 1 to 43
      PULSOUT 12, 650 'left
      PULSOUT 13, 750 'right
      PAUSE 20
    next
  return

turnRight:
  FOR x = 1 to 43
      PULSOUT 12, 750 'left
      PULSOUT 13, 850 'right
      PAUSE 20
    next
  return
```

My code with the c ool turn

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
TURNON CON 1
TURNOFF CON 0
PRESSED CON 0
LIFTED CON 1
'-----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'-----------------------------------
```

```
x VAR byte
GOSUB stopMove
GOSUB moveForward
GOSUB stopMove
GOSUB turnLeft
GOSUB stopMove
GOSUB moveForward
GOSUB stopMove
GOSUB turnRight
GOSUB stopMove
GOSUB moveForward
GOSUB stopMove
GOSUB moveBackward
END

moveBackward:
  FOR x = 1 to 100:
    PULSOUT 12, 650
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN

moveForward:
  FOR x = 1 to 100:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN

stopMove:
  PULSOUT 12, 750
  PULSOUT 13, 750
  PAUSE 1800
RETURN

turnLeft:
  FOR x = 1 to 23
    PULSOUT 12, 650
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN
```

```
turnRight:
  FOR x = 1 to 21
    PULSOUT 12, 850
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
TURN_ON CON 1
TURN_OFF CON 0
PUSHED CON 0
'------------------START INITIALIZATION
OUTPUT 11
initLed VAR OUT11
'LED FLASH
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'-----------------------------------
'Setting i/o  Ports and their names
OUTPUT 10
INPUT 3
led VAR OUT10
pushButton var IN3
```

```
'Main PRGM
checkButton: 'subRoutine
   IF pushButton = 1 THEN pushButton = PUSHED
     GOSUB turnOnLed
   ELSE
     GOSUB turnOffLed
   ENDIF
GOTO checkButton

'Subroutines here!
turnOnLed:
  led = TURN_ON
RETURN

turnOffLed:
  led = TURN_OFF
RETURN
```

i8i

```
ds'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
TURNON CON 1
TURNOFF CON 0
PRESSED CON 1
LIFTED CON 0
'-----------------------------------
'initialization
pushButton var IN3
OUTPUT 9
LED1 var OUT9
OUTPUT 8
LED2 var OUT8

GOSUB init
GOSUB checkButton

blinkLED:
   LED1 = TURNON
   LED2 = TURNOFF
   PAUSE 1200
   LED1 = TURNOFF
   LED2 = TURNON
   PAUSE 1200
   LED2 = TURNOFF
   RETURN

checkButton:
  DO
    DEBUG ? pushButton
    IF pushButton = PRESSED THEN
      GOSUB blinkLED
    ELSE
      PAUSE 200
    ENDIF
  LOOP
  RETURN

init:
  OUTPUT 11
  initLed VAR OUT11
  'blink the LEG
  initLED = TURNON
  PAUSE 500
  initLed = TURNOFF
  PAUSE 500
  FREQOUT 10, 1000, 2000
  RETURN
```

Move the heck forward:

```
DO
  PULSOUT 12, 850
  PULSOUT 13, 650
  PAUSE 20
LOOP
```

```
counter VAR Word
```

```
For counter = 1 TO 2
  PULSOUT 12, 650
  Pause 20
 NEXT

 For counter = 1 to 2
   pulsout 12, 750
   pause 20
next

for counter = 1 to 2
  pulsout 12, 850
  pause 20
 next

 end
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}

'constants
TURNON CON 1
TURNOFF CON 0

'------------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
```

```
'blink the LEG
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500

'beep the speaker
'1500ms = 1.5 seconds - 2000Hz note
FREQOUT 10,1500, 2000
'-----------------------------------

'Sample Text
counter VAR Word
do
 PULSOUT 12, 650
  pulsout 13, 850
  Pause 20
 loop




For counter = 1 TO 2
  PULSOUT 12, 650
  pulsout 13, 850
  Pause 20
 NEXT

 For counter = 1 to 2
   pulsout 12, 650
   pause 20
next

for counter = 1 to 2
  pulsout 13, 850
  pause 20
 next

 end
```

```
START
'{$STAMP BS2}
'{$PBASIC 2.5}

'constants
TURNON CON 1
TURNOFF CON 0

'--------------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
```

```
'blink the LEG
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500

FREQOUT 10, 1000, 2000
FREQOUT 10, 1000, 500
FREQOUT 10, 1000, 2000
FREQOUT 10, 1000, 500
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}

'constants
TURNON CON 1
TURNOFF CON 0

'------------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11

'blink the LEG
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500

FREQOUT 10, 1000, 2000

'--------------------------------
```

```
'Robotics with the Boe-Bot - HelloBoeBotYourTurn.bs2
'BASIC Stamp does simple math, and sends the results to the Debug Terminal
DEBUG CR, "What's 7 X 11?"
DEBUG CR, "The answer is: "
DEBUG DEC 7 * 11
END
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}

'constants
TURNON CON 1
TURNOFF CON 0

'-----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11

'blink the LEG
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000

'--------------------------------
'Robotics with the Boe-Bot - HelloBoeBotYourTurn.bs2
'BASIC Stamp does simpke math, and sends the results to the Debug Terminal
DEBUG DEC 1 + 2 + 3 + 4
DEBUG CR, "What's 7 X 11?", CR, "The answer is: ", 7 * 11
END
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}

'constants
TURNON CON 1
TURNOFF CON 0
'-----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LEG
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'--------------------------------
x VAR WORD
x = 65
DEBUG "x = ", DEC x, " in decimal", CR
DEBUG "x = ", x, " as an ASCII character", CR
DEBUG "x = ", BIN x, " in binary", CR
DEBUG "x = ", HEX x, " in hexadecimal", CR
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
TURNON CON 1
TURNOFF CON 0
'-----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LEG
initLED = TURNON
PAUSE 500
```

```
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'--------------------------------
x VAR WORD
DEBUG "Decimal..."
FOR x = 0 to 15
  DEBUG DEC x, " "
NEXT
DEBUG CR
DEBUG "Binary..."
FOR x = 0 to 15
  DEBUG BIN x, " "
NEXT
DEBUG CR
DEBUG "Hexadecimal..."
FOR x = 0 to 15
  DEBUG HEX x, " "
NEXT
```

3.

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
TURNON CON 1
TURNOFF CON 0
'-----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LEG
initLED = TURNON
```

```
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'-------------------------------
DEBUG 69, 114, 105, 99, 32, 80, 97, 110, 103, CR
DEBUG 83, 97, 109, 117, 101, 108, 32, 74, 101, 111, 110
```

Robotics with the BoeBot V2.2

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
TURNON CON 1
TURNOFF CON 0
'-----------------------------------
```

```
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LEG
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'--------------------------------

DEBUG "What's 1+2+3+4?"
DEBUG CR, "The answer is: "
DEBUG DEC 1+2+3+4
DEBUG 7 * 11

END
```

Lab #2 Code

```
DO
        DEBUG ? IN3
        PAUSE 250
LOOP
```

```
DO
        DEBUG ? IN3
        IF (IN3=1) THEN
                HIGH 9
                PAUSE 50
                LOW 9
                PAUSE 50
        ELSE
                PAUSE 100
        ENDIF
```

```
LOOP



'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
TURNON CON 1
TURNOFF CON 0
'-----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LEG
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000

'--------------------------------
DO
        DEBUG ? IN3
        IF (IN3=1) THEN
                HIGH 9
                PAUSE 1000
                LOW 9
                PAUSE 50
   HIGH 8
                PAUSE 1000
                LOW 8
                PAUSE 50
ELSE
                PAUSE 100
ENDIF
LOOP
```

```
DO
        DEBUG ? IN3
        IF (IN3 = 1) THEN
                HIGH 10
                PAUSE 50
                LOW 10
                PAUSE 50
        ELSE
                Pause 100
        ENDIF
LOOP
```

What value appears at Port 3 when pressed?	1
What value placed at Port 10 turns the LED on?	1

Your job is to rewrite the above code to look good using friendly names and comments. I have broken the code into sections with comments to help you. Yes, you must write in each empty cell/row.

‘declare constants and variables
TURNON CON 1 TURNOFF CON 0
PRESSED CON 1 LIFTED CON 0
empty line here for proper spacing of different code sections

```
‘set I/O ports and give friendly names
INPUT 3
pushButton = IN3
OUTPUT 10
initLED var OUT10


                              empty line here for proper spacing of different code sections
‘blink the led if the button is pushed
do
    DEBUG ? pushButton
    if pushButton = PRESSED then
        initLED = TURNON
        pause 50
        initLED = TURNOFF
        pause 50
     else
       pause 100
    endif
loop
```

HERE

```
‘blink the led if the button is pushed
do
    DEBUG ? pushButton
    If (pushButton = PUSHED) THEN
                led1 = TURN_ON
                 PAUSE 50
                    led1 = TURN_OFF
                 PAUSE 50
      ELSE
                  Pause 100
    endif
loop
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
TURNON CON 1
TURNOFF CON 0
PRESSED CON 1
LIFTED CON 0
'-----------------------------------
'initialization
pushButton var IN3
OUTPUT 9
LED1 var OUT9
OUTPUT 8
LED2 var OUT8

GOSUB initialization
GOSUB checkButton

blinkLED:
   LED1 = TURNON
   LED2 = TURNOFF
   PAUSE 1200
   LED1 = TURNOFF
   LED2 = TURNON
   PAUSE 1200
   LED2 = TURNOFF
   RETURN

checkButton:
  DO
    DEBUG ? pushButton
    IF pushButton = PRESSED THEN
      GOSUB blinkLED
    ELSE
      PAUSE 200
    ENDIF
  LOOP
  RETURN
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
TURNON CON 1
TURNOFF CON 0
PRESSED CON 0
LIFTED CON 1
'----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'----------------------------------

pushButton var IN3
OUTPUT 9
LED1 var OUT9
OUTPUT 8
LED2 var OUT8

GOSUB checkButton

checkButton:
   DEBUG ? pushButton
   IF pushButton = PRESSED THEN
     GOSUB turnOnLED
   ELSE
     GOSUB turnOffLED
   ENDIF
GOTO checkButton

turnOnLED:
  LED1 = TURNON
RETURN

turnOffLED:
  LED1 = TURNOFF
RETURN
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}

'declare constants/variables
CIRUIT_IS_HIGH CON 1
TURN_ON CON 1
TURN_OFF CON 0

'-----------------------------------------
'initialization
OUTPUT 9
initLed VAR OUT9

'blink the LED
initLed = TURN_ON
PAUSE 500
initLed = TURN_OFF
PAUSE 500
initLed = TURN_ON
PAUSE 500
initLed = TURN_OFF
PAUSE 500
initLed = TURN_ON
PAUSE 500
```

```
'beep the speaker
'1500ms = 1.5 seconds - 2000Hz note
FREQOUT 10,1500, 2000
'-------------------------------

'set I/O ports and give friendly names
INPUT 1
inputA VAR IN1
INPUT 2
inputB VAR IN2
INPUT 3
inputC VAR IN3

OUTPUT 4
led VAR OUT4



'main program
checkCircuitStatus: 'Note: The BS follow our order of operations (PNAXO)
  IF inputA AND (inputB OR inputC) = CIRUIT_IS_HIGH THEN
    GOSUB turnOnLed
  ELSE
    GOSUB turnOffLed
  ENDIF
GOTO checkCircuitStatus



'subroutines
turnOnLed:
  led = TURN_ON
RETURN

turnOffLed:
  led = TURN_OFF
RETURN
```

#Franklin version

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
TURNON CON 1
TURNOFF CON 0

'-----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'-----------------------------------

'set I/O ports and give friendly names
INPUT 0
inputA VAR IN0
INPUT 1
inputB VAR IN1
INPUT 2
inputC VAR IN2

OUTPUT 9
led VAR OUT9

'main program
checkCircuitStatus:
  IF inputA AND (inputB OR inputC) = CIRCUIT_IS_HIGH THEN
    GOSUB turnOnLed
  ELSE
    GOSUB turnOffLed
  ENDIF
GOTO checkCircuitStatus

'subroutines
turnOnLed:
  led = TURNON
RETURN

turnOffLed:
  led = TURNOFF
RETURN



'{$STAMP BS2}
```

```
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
TURNON CON 1
TURNOFF CON 0
PRESSED CON 0
LIFTED CON 1
'----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'----------------------------------
GOSUB stopMove
GOSUB moveForward

moveForward:
  x VAR byte
  FOR x = 1 to 55:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN

stopMove:
  PAUSE 2000
RETURN
```

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
TURNON CON 1
TURNOFF CON 0
PRESSED CON 0
LIFTED CON 1
'----------------------------------
'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
```

```
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
'----------------------------------
x VAR byte
GOSUB stopMove
GOSUB moveForward
GOSUB stopMove
GOSUB turnLeft
GOSUB stopMove
GOSUB moveForward
GOSUB stopMove
GOSUB turnRight
GOSUB stopMove
GOSUB moveForward
GOSUB stopMove
GOSUB moveBackward


moveBackward:
  FOR x = 1 to 160:
    PULSOUT 12, 650
    PULSOUT 13, 850
  NEXT
RETURN

moveForward:
  FOR x = 1 to 100:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN

stopMove:
  PULSOUT 12, 750
  PULSOUT 13, 750
  PAUSE 1800
RETURN

turnLeft:
  FOR x = 1 to 148
    PULSOUT 12, 650
    PULSOUT 13, 650
  NEXT
RETURN

turnRight:
  FOR x = 1 to 118
    PULSOUT 12, 850
    PULSOUT 13, 850
  NEXT
RETURN
```

Franklin:

```
'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
TURNON CON 1
TURNOFF CON 0
pulseCount VAR byte
EE_address VAR byte
instruction VAR byte

'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000

DATA "SFSLSFSRSFSB"

main:
  read EE_address, instruction
  EE_address = EE_address + 1
  if instruction = "F" then moveForward
  if instruction = "B" then moveReverse
  if instruction = "R" then turnRight
  if instruction = "L" then turnLeft
  if instruction = "S" then stopMove
END

moveForward:
  FOR pulseCount = 1 to 100:
    PULSOUT 12, 850
    PULSOUT 13, 650
    PAUSE 20
  NEXT
GOTO main

moveReverse:
  FOR pulseCount = 1 to 100:
    PULSOUT 12, 650
    PULSOUT 13, 850
    PAUSE 20
  NEXT
GOTO main
```

```
turnLeft:
  FOR pulseCount = 1 to 23
    PULSOUT 12, 650
    PULSOUT 13, 650
    PAUSE 20
  NEXT
GOTO main

turnRight:
  FOR pulseCount = 1 to 22
    PULSOUT 12, 850
    PULSOUT 13, 850
    PAUSE 20
  NEXT
GOTO main

stopMove:
  PULSOUT 12, 750
  PULSOUT 13, 750
  PAUSE 1000
GOTO main


/'{$STAMP BS2}
'{$PBASIC 2.5}
'constants
CIRCUIT_IS_HIGH CON 1
TURNON CON 1
TURNOFF CON 0
PRESSED CON 0
LIFTED CON 1
x VAR word
pulseCount VAR word
EE_address VAR byte
instruction VAR byte

'initialization
OUTPUT 11
initLed VAR OUT11
'blink the LED
initLED = TURNON
PAUSE 500
initLed = TURNOFF
'PAUSE 500
'FREQOUT 10, 1000, 1200
FREQOUT 10, 1000, 1500
'PAUSE 200
'FREQOUT 10, 700, 1200
'FREQOUT 10, 700, 1500
'PAUSE 200
```

```
'FREQOUT 10, 500, 1200
'FREQOUT 10, 500, 1500
'PAUSE 100
'FREQOUT 10, 300, 1200
'FREQOUT 10, 300, 1500
'PAUSE 100
'FREQOUT 10, 200, 1200
'FREQOUT 10, 200, 1500
'PAUSE 20
'FREQOUT 10, 100, 1200
'FREQOUT 10, 100, 1500
'PAUSE 20
'FREQOUT 10, 100, 1200
'FREQOUT 10, 100, 1500
'PAUSE 20
'FREQOUT 10, 100, 1200
'FREQOUT 10, 100, 1500
'PAUSE 20
'FREQOUT 10, 100, 1200
'FREQOUT 10, 100, 1500
'PAUSE 20
'FREQOUT 10, 2000, 1200

GOSUB moveForward
GOSUB stopMove
GOSUB turnRight
GOSUB moveForward
GOSUB stopMove
GOSUB turnRight
GOSUB moveForward
GOSUB stopMove
GOSUB turnRight
GOSUB moveForward
GOSUB stopMove
END

moveForward:
  FOR pulseCount = 1 to 100:
    PULSOUT 12, 750 + pulseCount
    PULSOUT 13, 750 - pulseCount
    PAUSE 20
  NEXT
RETURN

stopMove:
  FOR pulseCount = 1 to 100:
    PULSOUT 12, 850 - pulseCount
    PULSOUT 13, 650 + pulseCount
    PAUSE 20
  NEXT
RETURN
```

```
turnLeft:
  FOR pulseCount = 1 to 23
    PULSOUT 12, 650
    PULSOUT 13, 650
    PAUSE 20
  NEXT
RETURN

turnRight:
  FOR pulseCount = 1 to 23
    PULSOUT 12, 850
    PULSOUT 13, 850
    PAUSE 20
  NEXT
RETURN

moveReverse:
  PULSOUT 12, 750
  PULSOUT 13, 750
  PAUSE 1000
RETURN
```

```
  '{$STAMP BS2}
'{$PBASIC 2.5}
'variables
 x VAR word
 instruction VAR byte
 EE_address VAR byte
'constants
TURNON CON 1
TURNOFF CON 0
```

```
'-----------------------------------
'initialization
OUTPUT 9
initLED VAR OUT9
'blink the LED
initLED = TURNON
PAUSE 500
initLED = TURNOFF
PAUSE 500
FREQOUT 10, 1000, 2000
FREQOUT 10, 600, 2000
DATA "RLRLRL"
```

```
'----------------------------------
'main program
main:
  read EE_address, instruction
  EE_address = EE_address + 1
  if instruction = "F" then moveForward
  if instruction = "S" then stopMove
  if instruction = "B" then moveBackward
  if instruction = "L" then turnLeft
  if instruction = "R" then turnRight
  if instruction = "C" then circle
STOP

'subroutines
moveForward:
  FOR x=1 to 100
    PULSOUT 12,750 - x  'right
    PULSOUT 13,750 + x  'left
    PAUSE 20
  NEXT
GOTO main

stopMove:
FOR x=1 to 100
  PULSOUT 12, 650 + x
  PULSOUT 13, 850 - x
  PAUSE 20
NEXT
GOTO main

turnLeft:
  FOR x=1 to 190
    PULSOUT 12, 650
    PULSOUT 13, 773
    PAUSE 20
  NEXT
GOTO main

turnRight:
  FOR x=1 to 180
    PULSOUT 12,725
    PULSOUT 13,850
    PAUSE 20
  NEXT
GOTO main

moveBackward:
FOR x=1 to 120
  PULSOUT 12, 850
  PULSOUT 13, 650
  PAUSE 20
```

```
NEXT
GOTO main


circle:
FOR x=1 to 455
  PULSOUT 12, 725
  PULSOUT 13, 850
  PAUSE 20
NEXT
GOTO main
```

Photoresistor - light dependent resistor   (actually the photons of light)

- ***More light = less resistance***
- ***Less light = more resistance***

Considered analog device (infinite values)

Graph looks like a curved line // digital would be a linear straight line

Schematic symbol:
-~- a circle around a resistor with an arrow inside the circle pointing outside and 2 arrows outside of the circle pointing inside

Potentiometer (pot) - variable resistor (ANALOG)
A pot with “103” means - 10 with 3 additional zero’s meaning 0 - 1000 ohms

Middle pin = Common
Resistor on pin 1 to 2 (common) and 2(common) to pin 3 // kinda like a circuit

Direction you turn will increase or decrease the resistance of each side of the pins

Schematic:
-~- with an arrow pointing on the resistor

Ideal = make the robot to read a 1 or a 0
Forces the robot to read either a 1 or 0 (resistive divider schematic
ggfffsssr

Twinkle twinkle little star:

```
C7 CON          2093
G7 CON          3136
A7 CON          3520


FREQOUT 10,500, C7
  FREQOUT 10,500, C7
  FREQOUT 10,500, G7
  FREQOUT 10,500, G7
  FREQOUT 10,500, A7
  FREQOUT 10,500, A7
  FREQOUT 10,500, G7
```