```
'--------Title--------

' File......servo3.pbp
' Started....6/1/05
' Microcontroller used:  Microchip Technology 16F88
'                        microchip.com
' PicBasic Pro Code: micro-Engineering Labs, Inc.
'                    melabs.com

'--------Program Desciption--------

' Rotates servos into clockwise and counter-clockwise rotations,
' creating a panning motion.  Discussion about basic servo pulse
' control may be found at www.seattlerobotics.org/guide/servos.html or
' www.geocities.com/hobby_robotics/was.htm

'----------Related Lesson----------

' servo3.pbp is used in the lesson PIC PROGRAMMING 3 SERVOS at:
' http://cornerstonerobotics.
org/curriculum/lessons_year2/erii13_pic_programming3_servos.pdf

'------------Comments--------------

' WITH THE PIC16F88, MAKE SURE TO HAVE SEPARATE POWER
' SUPPLIES FOR THE PIC AND THE SERVO.  MAKE SURE TO
' HAVE A COMMON GROUND BETWEEN THE PIC AND SERVO.  We use one 9V
' battery and two 78L05 voltage regulators.  See
' discussion about voltage regulators at:
' http://cornerstonerobotics.
org/curriculum/lessons_year2/erii3_diodes_power_supplies_voltage_reg.pdf

' Also, initialize the state of PORTB as LOW
' since that will set the correct polarity of the
' PULSOUT statement.  See PULSOUT in PicBasic Pro
' Compiler manual by microEngineering Labs, Inc.
' The PicBasic Pro Compiler Manual is on line at:
' http://www.microengineeringlabs.com/resources/index.htm#Manuals
' Look around page 121 in the PicBasic Pro Compiler Manual

' Servos may be modified or hacked to allow
' for continuous rotation so they can be used
' as motors on small robots.  The book
' Amphibionics by Karl Williams gives an
' in depth treatment on how to modify servos  Also see:
' http://cornerstonerobotics.
org/curriculum/lessons_year2/erii17_hacking_servos.pdf

'-----------Connections-----------

'       PIC16F88 Pin            Wiring
'       ---------             ----------
'           RB0             Servo Control Wire
'           Vdd             +5 V
'           Vss             Ground
```

```
'           MCLR            4.7K Resistor to +5 V

'---------Revision History---------

' 11/14/07 Change MCU from 16F84A to 16F88
' 11/14/07 Add 16F88 oscillator initialization
' 11/27/07 Add power supply warning
' 5/21/08  Changed title from servo2.pbp to servo3.pbp

'------------Variables-------------

     p0     VAR     BYTE        ' Byte to store servo position

'----------Initialization----------

    PORTB = %00000000           ' Eqivalent to: PORTB = 0
                                ' Sets all PORTB pins to LOW(0 volts)
                                ' Make certain to include this
                                ' initialization as it sets the
                                ' proper polarity of pulses in
                                ' the PULSOUT command.

    OSCCON = $60                ' Sets the internal oscillator in the
                                ' 16F88 to 4 MHz

'-------------Main Code------------

start:

' Rotate counter-clockwise

    FOR p0 = 200 TO 100 STEP -1     ' Change value of pulse from 2 ms to
                                    ' 1 ms in steps of 10 us.  See next
                                    ' command.

    PULSOUT 0,p0                    ' Sends a pulse, p0, out on pin RB0.
                                    ' The period, p0, is multiplied by the
                                    ' increment for a 4 MHz oscillator
                                    ' (10 us) to get a pulse out time.
                                    ' For example, if p0 = 200,
                                    ' 200 * 10 us = 2000 us = 2 ms

    PAUSE 20 - p0/100               ' Pause 20 ms less pulse width (p0/100)
                                    ' If p0 = 200, p0/100 = 200/100 = 2 ms.
                                    ' This equation keeps the period of
                                    ' the servo pulse a constant 20 ms.

    NEXT p0                         ' Go back to the FOR statement and do
                                    ' next value of p0

'Rotate clockwise

    FOR p0 = 100 TO 200             ' Change value of pulse from 1 ms to
                                    ' 2 ms in steps of 10 us.
```

```
PULSOUT 0,p0                    ' Sends a pulse, p0, out on pin RB0.
                                ' Pulse out time varies from 1.0 msec
                                ' to 2.0 msec.

PAUSE 20 - p0/100               ' Pause 20 ms less pulse width (p0/100)

NEXT p0                         ' Go back to the FOR statement and do
                                ' next value of p0

GOTO start                       ' Makes the program run forever.

END
```