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


'------------------VARS--------------'
widthf VAR Word        'Forward Sensor Range
widthl VAR Word        'Left Sensor Range
widthr VAR Word        'Right Sensor Range
x      VAR Word        'Used for loops
milk   VAR Bit         'Button State
meat   VAR Bit         'Button State
bread  VAR Bit         'Button State


'------------------CONS--------------'
front_lim   CON 400    'For Sensor Range
right_angle CON 55     'Right Angle Pulse Number for the Servos


'------------------MAIN--------------'

GOSUB Button_Input
GOSUB Coordinator


'------------------SUBS--------------'

Button_Input:
  DO WHILE (IN9 <> 0)
  LOOP

  milk = 0 : meat = 0 : bread = 0

  DO WHILE (IN1 <> 0)
    IF (IN5 = 0) THEN
      milk = 1
    ELSEIF (IN4 = 0) THEN
      meat = 1
    ELSEIF (IN2 = 0) THEN
      bread = 1
    ENDIF
  LOOP
RETURN


Coordinator:
  IF (milk = 1 AND meat = 0 AND bread = 0) THEN
    GOSUB case1
  ELSEIF (milk = 0 AND meat = 1 AND bread = 0) THEN
    GOSUB case2
  ELSEIF (milk = 0 AND meat = 0 AND bread = 1) THEN
    GOSUB case3
  ELSEIF (milk = 1 AND meat = 1 AND bread = 0) THEN
    GOSUB case1_2
  ELSEIF (milk = 1 AND meat = 0 AND bread = 1) THEN
    GOSUB case1_3
```

```
  ELSEIF (milk = 0 AND meat = 1 AND bread = 1) THEN
    GOSUB case2_3
  ELSEIF (milk = 1 AND meat = 1 AND bread = 1) THEN
    GOSUB case1_2_3
  ENDIF
RETURN


case1:
  GOSUB forward
  GOSUB left
  GOSUB forward
RETURN


case2:
  GOSUB forward
  GOSUB forward
  GOSUB left
  GOSUB forward
RETURN


case3:
  GOSUB forward
  GOSUB forward
  GOSUB forward
  GOSUB forward
  GOSUB left
  GOSUB forward
RETURN


case1_2:
  GOSUB forward
  GOSUB left
  GOSUB forward

  DO WHILE (IN1 <> 0)
  LOOP

  GOSUB forward
  GOSUB right
  GOSUB forward
  GOSUB forward
  GOSUB right
  GOSUB forward
RETURN

case1_3:
  GOSUB forward
  GOSUB left
  GOSUB forward

  DO WHILE (IN1 <> 0)
  LOOP
```

```
  GOSUB forward
  GOSUB right
  GOSUB forward
  GOSUB forward
  GOSUB forward
  GOSUB right
  GOSUB forward
RETURN

case2_3:
  GOSUB FORWARD
  GOSUB FORWARD
  GOSUB LEFT
  GOSUB FORWARD

  DO WHILE (IN1 <> 0)
  LOOP

  GOSUB FORWARD
  GOSUB RIGHT
  GOSUB FORWARD
  GOSUB FORWARD
  GOSUB RIGHT
  GOSUB FORWARD
RETURN

case1_2_3:
  GOSUB forward
  GOSUB left
  GOSUB forward

  DO WHILE (IN1 <> 0)
  LOOP

  GOSUB forward
  GOSUB right
  GOSUB forward
  GOSUB forward
  GOSUB right
  GOSUB forward

  DO WHILE (IN1 <> 0)
  LOOP

  GOSUB forward
  GOSUB left
  GOSUB forward
  GOSUB forward
  GOSUB left
  GOSUB forward

  DO WHILE (IN1 <> 0)
  LOOP

  GOSUB forward
```

```
  GOSUB left
  GOSUB forward
  GOSUB forward
  GOSUB forward
  GOSUB forward
  GOSUB forward
  GOSUB left
  GOSUB forward
  GOSUB forward
  GOSUB left
RETURN


back:
  FOR x=1 TO 100
    GOSUB main_check
    PULSOUT 7,650     'Pin7 -> Left Servo
    PULSOUT 8,788     'Pin8 -> Right Servo
    PAUSE 16
  NEXT
  PAUSE 200
RETURN


forward:
  FOR x=1 TO 70
    GOSUB main_check
    PULSOUT 7,850
    PULSOUT 8,711
    PAUSE 16
  NEXT
  'PAUSE 200
RETURN


right:
  FOR x=1 TO right_angle
    PULSOUT 7,850
    PULSOUT 8,750
    PAUSE 20
  NEXT
  PAUSE 200
RETURN


left:
  FOR x=1 TO right_angle
    PULSOUT 7,750
    PULSOUT 8,650
    PAUSE 20
  NEXT
  PAUSE 200
RETURN
main_check:
  GOSUB front_sonar
  GOSUB left_sonar
```

```
  GOSUB right_sonar
  GOSUB check
RETURN


front_sonar:
  PULSOUT 11,5
  RCTIME 0,1,widthf
RETURN


left_sonar:
  PULSOUT 12,5
  RCTIME 3,1, widthl
RETURN


right_sonar:
  PULSOUT 6,5
  RCTIME 10,1,widthr
RETURN


check:
  IF (widthf < front_lim AND widthl < widthr) THEN
    GOSUB adjust_right
  ELSEIF (widthf < front_lim AND widthr < widthl) THEN
    GOSUB adjust_left
  ENDIF
RETURN

adjust_left:
  GOSUB left

  FOR x=1 TO 20
    PULSOUT 7,850
    PULSOUT 8,711
    PAUSE 20
  NEXT

  GOSUB right
RETURN

adjust_right:
  GOSUB right

  FOR x=1 TO 20
    PULSOUT 7,850
    PULSOUT 8,711
    PAUSE 20
  NEXT

  GOSUB left
RETURN
```