1.
    a. Know- The censors
    b. Show-Distance
    c. Do-

```
python
def obstacle_avoidance():
                left_sensor = read_left_sensor()
                center_sensor = read_center_sensor()
                right_sensor = read_right_sensor()

                if min([left_sensor, center_sensor, right_sensor]) < safe_distance:
                 if left_sensor < center_sensor and left_sensor < right_sensor:
                turn_right_slightly()
                elif right_sensor < center_sensor and right_sensor < left_sensor:
                turn_left_slightly()
                else:
                make_sharp_turn()
                 else:
                   move_forward()
```

2.
    a. Know-The sum
    b. Show-The count
    c. Do-

```
sum = 0
count = 0

while True:
 number = float(input("Enter a number (or 'done' to finish): "))
 if number == "done":
   break
 sum += number
count += 1
if count > 0:
 mean = sum / count
print("The mean of the entered numbers is:", mean)
else:
print("No numbers were entered.")
```

3.
    a. Know- The range
    b. Show- a and b
    c. Do-

```
def fibonacci(n):
   a, b = 0, 1
   print(a, b, end=" ")
   for i in range(2, n):
```

```
        c = a + b
        print(c, end=" ")
        a, b = b, c

# Get user input
num_terms = int(input("Enter the number of terms: "))

# Call the function to print the Fibonacci sequence
fibonacci(num_terms)
```

4.
    a. Know- Min and Max values
    b. Show- spread
    c. Do-

```
def calculate_spread(numbers):
    if not numbers:
        return "Cannot calculate spread on an empty list"

    min_value = min(numbers)
    max_value = max(numbers)
    spread = max_value - min_value

    return spread
```

5.
    a. Know- height
    b. Show- area
    c. Do-

```
wall_area = 2 * height * (length + width)
    ceiling_area = length * width
    total_area = wall_area + ceiling_area

    gallons_needed = (total_area / paint_coverage)
    gallons_needed = round(gallons_needed)

    total_cost = gallons_needed * paint_price

    return gallons_needed, total_cost
```

6.
    a. Know- 4 digit hexadecimals
    b. Show-decimal value
    c. Do-

```
decimal_value = 0
    power = 0
    hex_digits = {}
```

```
for digit in hex_num[::-1]:
    digit_value = int(digit) if digit.isdigit() else hex_digits[digit.upper()]
    decimal_value += digit_value * (16 ** power)
    power += 1

return decimal_value
```