The answer to question no. 1

```
Price_list = ["Litchi", 2.50, "Mango", 55.00, "Apple", 37.00, "Papaya", 33.00, "Watermelon", 55.00]
print(Price_list[0])
```

In this case, the print() function gives an output from the given price list. Even if there were 500 items in the list, the same print() functions would have given the same outcome as they are giving now.

The answer to question no. 2

I would have simply added -

```
Print(Price_list[6])
Print(Price_list[7])
```

The answer to the question no.3

```
Price_list = ["Litchi", 2.50, "Mango", 55.00, "Apple", 37.00, "Papaya", 33.00, "Watermelon", 55.00]
print(Price_list[0])
print(Price_list[1])
print(Price_list[2])
print(Price_list[3])
print(Price_list[4])
print(Price_list[5])
print(Price_list[6])
print(Price_list[7])
print(Price_list[8])
print(Price_list[9])
```

I have used 10 print() statements here

The answer to the question no. 4

This program prints

```
Litchi
2.5
Mango
55.0
Apple
37.0
Papaya
33.0
Watermelon
55.0
```

The answer to the question no. 5

i) For is used because it's called a **"For Loop"**

ii) **i** is used as a variable. We can use any letter in this place.

iii) We use the range function to simplify writing a for loop.

iv) 0 is the slot where we can put the number we want to count from the list. If we want to start printing from the very first item on the list, we put 0. That refers to the starting point or starting value of the loop. We need to put the exact number from where we expect our list to start printing.

v) 10 is the slot where we can put a number to command the end line. O we can say the stop value of the list. Or the stop value of the loop. For example, we used 10 as the stop value. Because we wanted to stop printing the loop right there.

The answer to the question no. 6

The for and the range function is so efficient. Because we can simplify many complicated kinds of stuff all in one line. We can determine the starting and the ending value altogether. We don't need to keep typing print statements over and over just to show our customized values.

The answer to the question no. 7

Because of the function – " Price_list(i+1) "

There's no specific space declared in the range sector. So, the same error keeps showing. It shows each item twice on the list. The outcome is not even organized properly.

The answer to the question no. 8

The difference between the output of verses 1 & 2 is the space declaration. The 1st verse is not well programmed. There were problems and it kept showing errors.

The second verse, it’s so well organized and well decorated with the required space. It shows the correct information along with the tagged items. That’s well-programmed code.

The answer to the question no. 9

The range function works better in this case because of the proper command line. The “2” on the second line shows the space between the string and the float. If we had 3 or more items, we would have taken that number in that place. That’s why, when we have added “ Price_list[i+1] “ in the last line, adding 2 on the range function is a must. And that is why it works so well.

The answer to the question no. 10

I have learned about many little parts with this assignment. I have learned proper facts about range function which were not crystal clear before this assignment. And the “Price_list[i+1] “ without the “.format” function was so vague. But, now it’s more clearer.