```
import csv
#Make use of the CSV file handling library

def read_csv_file(file_path):
    data = []
    with open(file_path, 'r') as csv_file:
        # Create a CSV reader object
        csv_reader = csv.reader(csv_file)
        # Iterate over each row in the CSV file
        for row in csv_reader:
            # Process each element in the row
            processed_row = []
            for element in row:
                # Try to convert element to a float (currency)
                try:
                    element = float(element)

                except ValueError:
                    pass  # If it fails, keep original string

                # Try to convert element to an integer
                try:
                    element = int(element)

                except ValueError:
                    pass  # If it fails, keep the original string

                processed_row.append(element)

            # Add the processed row to the data list
            data.append(processed_row)

    return data


# Example usage:
file_path = 'C:\\Temp\\Task4a_data.csv' # Assuming CSV downloaded
csv_data = read_csv_file(file_path)
print(csv_data)
```

This example defines a read_csv_file function that takes the file path as an argument and returns a list of lists containing the processed data. The function uses the csv.reader object to iterate through the rows in the CSV file. For each element in a row, it attempts to convert the element to a float (for fixed-point numbers) and then to an integer. If both conversions fail, it keeps the original string.

Make sure to replace 'your_file.csv' with the actual path to your CSV file when using this code. Additionally, you may need to adjust the conversion logic based on the specific format of your CSV file.

To process the data you can now make use of the csv_data list (a data structure now held in memory).

```
# Assuming csv_data is obtained from the previous example

def calculate_total_weight(data):
    total = 0

    for row in data:
        # The third element [2] in each row is KG purchased
        try:
            total += int(row[2])

        except ValueError:
            print(row[1])

    return total

# Example usage:
total_value = calculate_total_weight(csv_data)
print("Total weight of all items purchased:", total_value,"Kg")
```

If you just wanted to total up the weight of oranges purchased then you could do it this way:

```
    for row in data:
        # Assuming the third element is definitely an integer
        if row[0] == "Oranges":
            total += row[2]

    return total
```

Instead of the literal string “Oranges” you could have a string variable which holds the name of the product that the user has selected/entered.

The CSV file provided in task4a (ESP 1) contains the following elements in each row[] of data:

[0] Product, [1] Supplier (not in the task 3 data), [2] KGs Purchased, [3] Purchase Price, [4] KGs Sold, [5] Selling Price, and [6] Date

Product is a string, KGs are integers, Prices are float and Date is a datetime.