```
# Given DataList
DataList = [25, 28, -11, 0, 29, 10, -4, 45, 19, -8, 7, 10, 3, 20, 12, -4, -11, 20, 31]
# a) Remove all duplicate elements without creating a new list
DataList = list(dict.fromkeys(DataList))  # Removes duplicates while preserving order
print("Modified DataList without duplicates:", DataList)
# b) Identify the 6th smallest and 2nd largest numbers
sorted_list = sorted(DataList)
sixth_smallest = sorted_list[5]
second_largest = sorted_list[-2]
print("6th Smallest:", sixth_smallest)
print("2nd Largest:", second_largest)
# c) Rotate the elements of the modified DataList by 3 positions to the left
n = 3  # Number of positions to rotate
DataList = DataList[n:] + DataList[:n]  # Rotate left without extra variable
print("DataList after 3-position left rotation:", DataList)
# d) Divide the modified DataList into two lists: EvenList and OddList
EvenList = [num for num in DataList if num % 2 == 0]
OddList = [num for num in DataList if num % 2 != 0]
print("EvenList:", EvenList)
print("OddList:", OddList)
```

2.

```
# Given lists
Product_IDs = [1001, 1002, 1003, 1004, 1005]
Prices = [250, 450, 300, 800, 150]
# a) Generate a dictionary ProductPriceDict by mapping Product_IDs to Prices
ProductPriceDict = dict(zip(Product_IDs, Prices))
print("ProductPriceDict:", ProductPriceDict)
# b) Find the price of the most expensive product without using any built-in functions
max_price = -1  # Initialize with a very low value
for price in Prices:
    if price > max_price:
        max_price = price
print("Price of the most expensive product:", max_price)
# c) Write a function to get the price of a product by its ID
def get_price_by_id(product_id):
    if product_id in ProductPriceDict:
        return ProductPriceDict[product_id]
    else:
        return "Invalid Product ID"
# Example usage of the function
product_id_to_check = 1004
print(f"Price of Product ID {product_id_to_check}:", get_price_by_id(product_id_to_check))
# d) Function to validate a product ID and return its price or an error message
def validate_and_get_price(product_id):
    return ProductPriceDict.get(product_id, "Invalid Product ID")
# Example usage of the function
product_id_to_check = 1006
print(f"Validation result for Product ID {product_id_to_check}:", validate_and_get_price(product_id_to_check))
```