```
"""
Este módulo calcula el total de ventas a partir de un catálogo de precios y un
registro de ventas, ambos proporcionados en archivos JSON. Calcula el total de
ventas, maneja datos inválidos y reporta el tiempo total de ejecución del
cálculo.
"""

import json
import sys
import time


def read_json_file(filename):
    """
    Lee y devuelve el contenido de un archivo JSON.

    Args:
        filename (str): La ruta al archivo JSON a leer.

    Returns:
        dict/list: Contenido del archivo JSON.
    """
    try:
        with open(filename, 'r', encoding='utf-8') as file:
            return json.load(file)
    except FileNotFoundError:
        print(f"Error: El archivo {filename} no fue encontrado.")
        return None
    except json.JSONDecodeError:
        print(f"Error: Archivo {filename} no es válido JSON.")
        return None


def calculate_sales(prices, sales):
    """
    Calcula el total de ventas basado en los precios y el registro de ventas.
    Args:
        prices (dict): Un diccionario con los precios de los productos.
        sales (list): Lista de registros de ventas.
    Returns:
        float: El total de las ventas.
    """
    total_sales = 0
    products_not_found = set()  # Guarda los productos no encontrados
    for sale in sales:
        product_title = sale.get('Product')
        quantity = sale.get('Quantity', 0)
        if product_title in prices:
            total_sales += prices[product_title] * quantity
        else:
            products_not_found.add(product_title)
            # Añade el producto no encontrado

    # Imprime los productos no encontrados
    if products_not_found:
        print("Productos no encontrados en el catálogo:")
        print(", ".join(products_not_found))

    return total_sales


def main(product_list_filename, sales_filename):
    """
    Función principal que ejecuta el cálculo del total de ventas.

    Lee los archivos de lista de productos y registros de ventas, calcula el
```

```
    total de ventas combinando estos datos, e imprime y guarda el resultado.

    Args:
        product_list_filename (str): Ruta al archivo JSON de la lista de
                                     productos.
        sales_filename (str): Ruta al archivo JSON de los registros de ventas.
    """
    start_time = time.time()

    product_list = read_json_file(product_list_filename)
    if product_list is None:
        return

    sales_record = read_json_file(sales_filename)
    if sales_record is None:
        return

    prices = {item['title']: item['price'] for item in product_list}

    total_sales = calculate_sales(prices, sales_record)

    elapsed_time = time.time() - start_time

    result_string = (f"Total ventas: ${total_sales:.2f}\n"
                     f"Tiempo ejecución: {elapsed_time:.2f}s")
    print(result_string)

    with open('sales_results.txt', 'w', encoding='utf-8') as file:
        file.write(result_string)


if __name__ == '__main__':
    if len(sys.argv) != 3:
        print("Uso: python compute_sales.py product_list.json "
              "sales_record.json")
    else:
        main(sys.argv[1], sys.argv[2])
```