Building a Contact Management System

Here's a comprehensive assignment that covers various aspects of Python's basic syntax, string manipulation, numeric data types, conversion functions, and the `print()` function.

Assignment:

Task: Building a Contact Management System

Create a simple contact management system where users can add, view, and search for contacts. Each contact should have a name, phone number, and email address.

Requirements:

1. Implement a menu-driven system with the following options:
   - Add a new contact.
   - View all contacts.
   - Search for a contact by name.
   - Exit the program.

2. For adding a new contact:
   - Prompt the user to enter the contact's name, phone number, and email address.
   - Ensure that the phone number is a valid numeric value.

3. For viewing all contacts:
   - Display a formatted list of all contacts, including their names, phone numbers, and email addresses.
   - If the contact list is empty, print a message indicating that there are no contacts.

4. For searching for a contact by name:
   - Prompt the user to enter a name to search for.
   - Display the details of the contact if found, or indicate if the contact is not in the list.

5. Implement proper error handling to handle cases such as invalid numeric input for phone numbers or incorrect menu selections.

6. Use appropriate string formatting for displaying contact information.

7. Ensure that the program runs continuously until the user chooses to exit.

Sample Answer:

```python
contacts = []
```

```
while True:
    print("\nContact Management System:")
    print("1. Add a new contact")
    print("2. View all contacts")
    print("3. Search for a contact by name")
    print("4. Exit")

    choice = input("Enter your choice (1-4): ")

    if choice == "1":
        name = input("Enter the contact's name: ")
        phone = input("Enter the contact's phone number: ")
        while not phone.isdigit():
            print("Invalid phone number. Please enter a numeric value.")
            phone = input("Enter the contact's phone number: ")
        email = input("Enter the contact's email address: ")
        contacts.append({"Name": name, "Phone": phone, "Email": email})
        print("Contact added successfully!")

    elif choice == "2":
        if not contacts:
            print("No contacts available.")
        else:
            print("\nAll Contacts:")
            for contact in contacts:
                print(f"Name: {contact['Name']}, Phone: {contact['Phone']}, Email:
{contact['Email']}")

    elif choice == "3":
        search_name = input("Enter the name to search for: ")
        found_contact = None
        for contact in contacts:
            if contact["Name"].lower() == search_name.lower():
                found_contact = contact
                break
        if found_contact:
            print("\nContact Found:")
            print(f"Name: {found_contact['Name']}, Phone: {found_contact['Phone']}, Email:
{found_contact['Email']}")
        else:
            print(f"No contact found with the name '{search_name}'.")

    elif choice == "4":
        print("Exiting the Contact Management System. Goodbye!")
        break

    else:
        print("Invalid choice. Please enter a number between 1 and 4.")
```

This assignment integrates concepts such as user input, string manipulation, numeric validation, list manipulation, and error handling to create a basic contact management system. Feel free to adapt and modify the code based on your preferences or to add more features.