Nombre y Apellidos:

Considere el siguiente código:

```
class DB {
private:
  Persona* personas;  // Puntero a un arreglo dinámico de Persona
  int size,last;

public:
  DB(int n)  {
      size = n;
      personas = new Persona[size];  // Asignar memoria para el arreglo
      last = 0;
  }

  void add(Persona p) {
    if (last >= 0 && last < size) {
      personas[last] = p;
      last++;
    }else{
      throw std::runtime_error("Error de índice");
    }
  }

  void mostrarRegistros() {
    for (int i = 0; i < last; i++) {
        std::cout << "Registro " << i << ": " << personas[i].toString() << "\n";
    }
  }
};
```

Generalice el uso de esta clase a través del uso de templates.

Solución:

```
template <typename Q>
class DB {
private:
  Q* datos;  // Puntero a un arreglo dinámico de objetos de clase Q
  int size,last;

public:
  DB(int n)  {
      size = n;
      datos = new Q[size];  // Asignar memoria para el arreglo
      last = 0;
  }

  void add(Q p) {
    if (last >= 0 && last < size) {
      datos [last] = p;
      last++;
    }else{
      throw std::runtime_error("Error de índice");
    }
  }

  void mostrarRegistros() {
    for (int i = 0; i < last; i++) {
        std::cout << "Registro " << i << ": " << datos[i].toString() << "\n";
    }
  }
};
```

Ejemplo de uso de esta clase template.

Se asume la existencia de las clases Persona, Casa y Auto.

```
// Instanciar la clase DB para el objeto Persona
DB<Persona> dbPersonas(200);
// Instanciar la clase DB para el objeto Casa
DB<Casa> dbCasitas(400);
// Instanciar la clase DB para el objeto Auto
DB<Auto> dbAutitos(1200);
```