Resolución Examen de Entrada - UNSA
Tecnología de Objetos

Richart Escobedo
Universidad Nacional de San Agustín de Arequipa

1 de septiembre de 2025

1. Parte 1: Creación de clases en C++

1.1. Código de la clase student.cpp

Este archivo define una clase para representar a un estudiante con CUI y nombres, como lo pide el examen.

```cpp
#include <iostream>
#include <string>

class Student {
public:
    // Atributos de la clase
    std::string names;
    std::string cui;
    std::string paternal_surname;
    std::string maternal_surname;

    // Constructor para inicializar el objeto
    Student(std::string p_cui, std::string p_paternal,
            std::string p_maternal, std::string p_names) {
        cui = p_cui;
        paternal_surname = p_paternal;
        maternal_surname = p_maternal;
        names = p_names;
    }

    // Método para mostrar la información del estudiante
    void display_info() {
        std::cout << cui << "-" << paternal_surname << "/" <<
        maternal_surname << ", " << names << std::endl;
    }
};
```

1.2. Código del programa me.cpp

Este programa utiliza la clase `Student` para mostrar la información del estudiante en la terminal.

```cpp
#include "student.cpp"

int main() {
    // Reemplace los valores con su propia información
    Student me("CUI-DEL-ESTUDIANTE", "APELLIDO_PATERNO",
               "APELLIDO_MATERNO", "NOMBRES");

    // Llamamos al método para mostrar la información
```

```
    me.display_info();

    return 0;
}
```

2. Parte 2: Compilación y ejecución en la terminal

2.1. Provisión de un compilador C++

Para proveer un compilador C++ al sistema operativo:

- **MS Windows**: Se instala MinGW-w64, que proporciona el compilador `g++`.
- **GNU/Linux**: Se instala el paquete `build-essential` a través del gestor de paquetes. Por ejemplo, en distribuciones basadas en Debian (como Ubuntu), se usa el comando `sudo apt-get install build-essential`.

2.2. Comandos para compilar y ejecutar

Suponiendo que los archivos están en el directorio actual, los comandos son:

- **Compilación:** Para generar un ejecutable con el nombre `me.out` o `me.exe`:

  ```
  g++ me.cpp student.cpp -o me.out
  ```

- **Ejecución:** Para ejecutar el programa desde la terminal:

  ```
  ./me.out
  ```

3. Parte 3: Definición de la función 'invertir()'

A continuación se presenta la definición de la función `invertir()` que invierte un número entero positivo.

```
#include <iostream>

// Función que invierte un número entero positivo
int invertir(int numero) {
    int nuevo_numero = 0;
    while (numero > 0) {
        int digito = numero % 10;
        nuevo_numero = nuevo_numero * 10 + digito;
        numero = numero / 10;
    }
    return nuevo_numero;
}
```

4. Parte 4: Preguntas teóricas

4.1. ¿Qué son los estándares de codificación?

Los estándares de codificación son un conjunto de reglas y convenciones para escribir código fuente. Su objetivo es mejorar la legibilidad y la consistencia, lo que facilita el trabajo en equipo.

4.2. ¿Quién es considerado el padre de la Programación Orientada a Objetos?

El padre de la Programación Orientada a Objetos (POO) es Alan Kay, quien fue un pionero en este campo y desarrolló el lenguaje Smalltalk.

4.3. ¿Qué diferencia hay entre un editor de texto plano y un editor de texto enriquecido?

Un editor de texto plano trabaja solo con texto sin formato, sin posibilidad de aplicar estilos. Un editor de texto enriquecido permite dar formato al texto, como negritas, cursivas o diferentes fuentes.

4.4. ¿Cuál usar (TABS o ESPACIOS) para la identación de sus programas?

Aunque es una preferencia personal, la mayoría de los estándares de la industria recomiendan usar espacios para la indentación. Esto asegura una indentación consistente en cualquier editor y entorno, lo que es crucial para la colaboración en proyectos de programación.