Escuela Profesional de Ciencia de la Computación

Curso: Lenguajes de Programación

2024-II

NOMBRE: CESAR LENGUA MÁLAGA

1. **Revisar el siguiente código en C, que imprime y por qué?**

```
int main () {
   int a, b, c;  // se crean tres enteros
   b = c = 10;  // b y c tienen el valor de 10
   a = b++ + b++; //(1)  -> a se va a igualar a la operación de b++ + b++. En el primer b++ se incrementa a 11 pero va a devolver 10 antes del incremento, en el segundo b++ se incrementa a 12 pero va a devolver 11 antes del incremento. Para entender es que al hacer b++ el incremento se realizará después de "ejecutar b++" pero este va a devolver el valor antes del incremento:
   a = 10 + 11, b = 12
   printf("%d\n", a);  // a imprime 21
   printf("%d\n", b);  // b imprime 12

   a= ++c + ++c; //(2)  -> en este caso el ++c se hace antes por lo que el valor cambia y lo más importante es que RETORNA EL VALOR NUEVO, en este caso sería a = 11+12 = 23
   printf("%d\n", a); // a imprime 23
   printf("%d\n", c);  // c imprime 12
   b = 10;

   a = b++ + b; //(3)  // en este caso solo ocurre un incremento por lo que b tendrá el nuevo valor de 11 pero en el b++ retornará el valor antes de su incremento por lo que es 10, toda la operación sería a = 10 + 11 = 21
   printf("%d\n", a);  // a imprime 21
   printf("%d\n", b);  // b imprime 11
   a = 10;
   b = 5;

   if (a>b || ++b>5) //(4)  -> ambos operando son verdaderos pero primero
```

```
    se evalúa el a>b como es verdadero ya no se evalúa el segundo
    operando, si en caso el primer operando sea falso, pasaría al segundo
    operando y si se incrementaría b.
    printf("%d\n", b);  // b imprime 5
    a = 1;
    b = 5;

    if (a>b || ++b>5) //(5) -> el primer operando es falso por lo que pasa a
    comprobar el otro y si es verdadero porque b incrementa a 6.
    printf("%d\n", b);  // b imprime 6
    return 0;
}
```

2. Que realiza el siguiente código y cual es el resultado de la ejecución:

en primer lugar el código no se podría ejecutar ya que no tiene el using namespace std (esto no podría hacer cout se tendría hacer std::cout) ya que puedo entender que se está ejecutando en c++ por el <iostream>

```
#include <iostream>
class base {
        public:
        virtual void mostrar1() {  -> es virtual, puede ser sobrescrito por las
        clases derivadas
                cout << "base 1\n";}
        void mostrar2 () {
                cout << "base 2 \n";}
};
class derivada1: public base {
        public:
        void mostrar1() {
                cout << "derivada 1\n";}
};

class derivada2: public base {
        public:
        void mostrar2 () {
                cout << "derivada 2 \n";}
};
```

```
void prt(base *q) {  -> llama a mostrar1() y mostrar2()  sobre el objetobase
        q->mostrar1();
        q->mostrar2();
}
void main( ) {  -> se me hace raro el void main(), pienso que debería ser int
main()

        base b;  // b objeto de la clase base

        base *p;  // p puntero de la clase base

        derivada1 dv1;  // dv1 objeto de la clase derivada1

        derivada2 dv2;  // dv2 objeto de la clase derivada 2

        p = &b;  // apunta a la dirección de memoria del objeto base

        prt(p);  // llama a mostrar1()  y mostrar2() sobre el objeto base por
        lo que su salida es: base1 y base2

        dv1.mostrar1();  //  imprime derivada1

        p = &dv1;  // apunta a la dirección de memoria del objeto
        derivada1

        prt(p);  // como mostrar1() es virtual, se llama a la función
        sobrescrita en derivada1 pero mostrar2() no es virtual por lo que se
        llama a la versión de la clase base por lo que imprime: derivada 1 y
        base2

        dv2.mostrar2(); // imprime derivada2

        p = &dv2;  // apunta a la dirección del objeto derivada2

        prt(p);   // en este caso mostrar1() es virtual pero en la clase
        derivada2 no está sobrescribiendo dicha función por lo que llama a
        la versión de la clase base. mostrar2() si se encuentra sobrescrito
        en la clase derivada2 pero no es virtual así que llama a la versión de
        la clase base por lo que imprime: base1 y base2
}
```