Nombre: _____

Certamen 2 ELO-329 Diseño y Programación Orientada a Objetos (1s2023)

En este certamen usted no podrá hacer preguntas. Si algo no está claro, indíquelo en su respuesta, haga una suposición razonable y resuelva conforme a ella.

Primera parte, **sin apuntes** (40 minutos):
Primera pregunta (30 pts.)

A. Complete en el siguiente diagrama los nombres de al menos 5 de los elementos que componen el stack Android.

<table>
<tr><td>Aplicaciones del sistema</td><td>Aplicaciones de usuario</td></tr>
<tr><td colspan="2">API del Framework Java</td></tr>
<tr><td>Bibliotecas nativas C y C++</td><td>Android Runtime</td></tr>
<tr><td colspan="2">Capa de abstracción del Hardware</td></tr>
<tr><td colspan="2">Kernel de Linux</td></tr>
</table>

B. En el código proporcionado, se han declarado varias funciones miembro dentro de la clase MiClase con diferentes modificadores const.
Indique cuáles de estas funciones miembro generarán errores de compilación y cuáles no. Para aquellas que generen errores, justifique su respuesta.

```
class MiClase{
   public:
      void add1(int *pinput, int x);
      void add2(int *pinput, int x) const;
      void add3(int *const pinput, int x);
      void add4(const int *pinput, int x);
      void add5(const int *const pinput, int x) const;
   private:
      int *px = new int(5);
};
```

```
void MiClase::add1(int *pinput, int x){
   this->px += *pinput + x;
   std::cout << *px << std::endl;
}
//R:
```

```
~~void MiClase::add3(int *const pinput, int x){~~
   ~~this->px += *pinput + x;~~
   ~~std::cout << *px << std::endl;~~
~~}~~
~~//R:~~
```

Nombre: _____

No genera errores

```
void MiClase::add2(int *pinput, int x) const{
    this->px += *pinput + x;
    std::cout << *px << std::endl;
}
//R:
```

add2 genera error debido al calificador const al final de la definición de dicha función miembro. Este calificador impide que se modifiquen los atributos de una clase, que es justamente lo que se intenta hacer en add2

```
void MiClase::add3(int *const pinput, int x){
    this->px += *pinput + x;
    std::cout << *px << std::endl;
}
//R:
```

No genera errores

```
void MiClase::add4(const int *pinput, int x){
    this->px += *pinput + x;
    std::cout << *px << std::endl;
}
//R:
```

No genera errores

```
void MiClase::add5(const int *const pinput, int x) const{
    this->px += *pinput + x;
    std::cout << *px << std::endl;
}
//R:
```

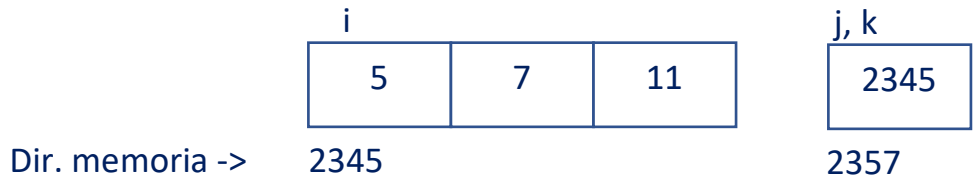
add5 genera error debido al calificador const al final de la definición de dicha función miembro (mismo motivo que add2).

C. Sabiendo que el compilador ubica la variable **i** en la dirección 2345 y **j** en la dirección 2357 ¿qué debería imprimir el siguiente código?

```
int i[3] = {5, 7, 11};
int * j = i;
int * &k = j;
cout << "i =" << i << endl;                    // →
cout << "&j =" << &j << endl;                  // →
cout << "k =" << k << endl;                    // →
cout << "*k =" << *k << endl;                  // →
cout << "*(j+2)=" << *(j+2) << endl;           // →
```

Aquí ponga sus respuestas
i =2345
&j =2357
k =2345
***k =5**
***(j+2)=11**

Nombre: _____



D. Considerando las declaraciones de clases dadas, en columna “Con virtual” indique qué aparece por consola al ejecutar el código de la derecha.

Si eliminamos la palabra virtual en el método foo() de cada clase, en columna “Sin virtual” indique qué aparece por consola al ejecutar el código de la derecha.

```
class A {
   public:
      virtual void foo() { cout << “A” << endl; }
};
class B: public A {
   public:
      virtual void foo() { cout << “B” << endl; }
};
class C: public B {
   public:
      virtual void foo() { cout << “C” << endl; }
};
```

```
B b;
A *a=&b;
a ->foo();          // →
C c;
A aa=c;
aa.foo();           //→
a=&c;
B *bb = (B*) a;
bb->foo();          // →
```

Con virtual	Sin virtual
B	**A**
A	**A**
C	**B**

E. En el siguiente ejemplo, se tienen las tres funciones: ‘funcionA’, ‘funcionB’ y la función principal ‘main’. Al ejecutar la función main mostrada, índique qué mensaje se muestra al ingresar por consola el valor negativo **-2** y qué se muestra al ingresar el valor positivo **3**.

```
void funcionA(int valor) {
   if (valor < 0)
        throw "El valor no puede ser negativo";
   cout << "El valor es: " << valor << endl;
}
void funcionB(int valor) {
    try {
         funcionA(valor);
    } catch (const char* mensaje) {
        cout << "Se produjo una excepción: ";
        cout << mensaje << endl;
    }
}
```

```
int main()  {
    int valor;
    cin >> valor;
     try {
        funcionB(valor);
     } catch (...) {
        cout << "Se produjo una excepción";
        cout<< "desconocida" << endl;
    }
}
```

R:

Nombre: _____

- A: “Se produjo una excepción: El valor no puede ser negativo”.
- B: “El valor es: 3“.

F. Mencione 1 semejanza y 1 diferencia entre casos de uso e historias de usuario.

R:

- Semejanza: Se utiliza para el levantamiento de requerimientos.
- Diferencia: Caso de uso es más detallado, Historia de Usuario es menos precisa.