

Examen de Metodología de la Programación II
Curso 2004–2005
Convocatoria de Febrero.

Sección informativa:

- La duración del examen será de 2'5 horas.
- Cada folio llevará Apellidos, Nombre, DNI y Grupo (Gestión/Sistemas)
- Cada ejercicio se entregará en folios distintos.
- Sólo se podrán realizar consultas durante los primeros 30 minutos.

Ejercicio 1.-

Una empresa paga a sus empleados mensualmente. La compañía tiene tres tipos de empleados: *encargados* que reciben un salario semanal fijo, sin importar el número de horas trabajadas; *subcontratados*, que reciben un sueldo por el número de horas trabajadas; *vendedores* que cobran por horas más una comisión de las ventas realizadas. Para ello debemos implementar una función llamada `calcular_pago` que calcule lo que hay que pagar cada mes a cada empleado. Se pide:

- (a) Dibujar el diseño de la jerarquía de clases. **0'5 puntos**
- (b) Implementar en JAVA el código de las correspondientes clases. **2 puntos**

Ejercicio 2.-

Dado el siguiente algoritmo siguiente calcular el número de operaciones en el *mejor* y en el *peor* de los casos. **2'5 puntos**

```
void algoritmo1(int a[], int n) {
    int j, cont, max = 0;
    for (int i=0;i<n;i++)
    {
        cont=1;
        j=j+1;
        while (a[i]<=a[j])
        {
            j=j+1;
            cont=cont+1;
        }
        if (cont > max)
            max = cont;
    }
}
```

Apellidos Nombre

Ejercicio 3.-

2 puntos

Busca los errores que existen en el siguiente código C++ y reescríbelo correctamente:

```
class base { |
    private: |
        int p; |
    public: |
        void base(int cuanto) { |
            p=cuanto;} |
        void ~base(void){} |
}; |

class derivada: public base { |
    protected: |
        int p2; |
    public: |
        void derivada(int cuanto) { |
            p=cuanto; |
            p2=p+1; |
        } |
        void ~derivada(void){} |
}; |

int main(void) { |
    base unaclase=new derivada(5); |
    unaclase.p2=unaclase.p2+1; |
    return 0; |
}
```

Apellidos Nombre

Ejercicio 4.-

Completa las siguientes frases referentes a la programación en C++:

3 puntos

- Una función no miembro de una clase debe declararse como [] de una clase para tener acceso a los datos miembro de dicha clase.
- El operador [] asigna la memoria de manera dinámica para un objeto de tipo específico y devuelve un [] de dicho tipo.
- Un dato miembro [] representa información propia de la clase.
- Las funciones miembro no [] tienen acceso a un puntero a sí mismo llamado [].
- El operador [] libera la memoria previamente asignada con `new`.
- La palabra clave [] introduce la definición de una estructura.
- Se accede a los miembros de una clase mediante el operador [] junto con el nombre de un objeto (o la referencia a un objeto) de la clase, o mediante el operador [] junto con un puntero a un objeto de la clase.
- Los miembros de una clase especificados como [] son accesibles para las funciones miembro de la clase y amigas de la clase.
- Al conjunto de funciones miembro públicas de una clase se les llama [] de la clase.
- Los miembros de una clase especificados como [] son accesibles en cualquier parte dentro del alcance del objeto de la clase.