Nombre: _____ **Sección:** _____

EXAMEN FINAL

¡Anota tu nombre y número de sección en <u>todas las hojas</u> del examen AHORA! (penalidad de 5 puntos)

Tienes 2 horas para completar tres problemas. Lee cuidadosamente todo el examen antes de empezar a trabajar. Muestra todo el trabajo conducente a tu contestación. Podrás recibir crédito parcial por contestaciones parciales siempre y cuando muestres tu trabajo por escrito. Utiliza la parte trasera de las páginas para cómputos. Usa tu tiempo inteligentemente. Exito!

ICOM 4015 Staff

1	35
2	20
3	30
4	15
Total	100

Nombre: _____ **Sección:** _____

Problema 1. Data Abstraction - Los BigInt's contraatacan (35 Points)

In the spirit of BigInt's, let's assume you want to represent huge integers with up to 12 decimal digits using dynamic arrays of 12 integer digits. For example, the number one thousand four hundred twenty two (1,422) will be represented with the array

{ 0, 0 ,0, 0, 0, 0, 0, 0, 1, 4, 2, 2 }

Here is what a class implementing huge integers may look like:

```
class HugeInt {
private:
    int* digits;
public:
    HugeInt();
    HugeInt(n);
    ~HugeInt();
    HugeInt add(HugeInt);

    // Additional methods omitted
}
```

a. (18 points) Complete the definition of the followng constructors and the destructor

```
HugeInt::HugeInt(int n){
    // Creates a new HugeInt and sets its value to n









}
//Copy Constructor
HugeInt::HugeInt(const HugeInt& n){
    // Creates a new HugeInt as a copy of n









}
HugeInt::~HugeInt(){



}
```

Nombre: _____ **Sección:** _____

b. (17 points) Implement a method that adds a HugeInt to the target object and returns the resulting HugeInt. If the resulting HugeInt has more than 12 digits, the method should return a HugeInt representing a zero.

```
HugeInt HugeInt::add(const BigInt& b)  {









}
```

Nombre: _____ **Sección:** _____

Problema 2 (20 Points)

Parte A. Contesta las siguientes preguntas concisamente:

1. (5 pts) ¿Qué significa que un método sea declarado "virtual"?

2. (5 pts) ¿Qué es una clase abstracta y para qué puede ser útil?

ICOM4015 Otoño 2000 Exámen Final 19/diciembre/2000

Nombre: _____ **Sección:** _____

Parte B. (10 pts) Escriba el output de este programa:

```
class Auto {
protected:
   float Precio;
   char* Color;
public:
   Auto(float P, char* C) {
       Precio=P;
       strdup(Color,C);
   }
   Auto() {}
   ~Auto() {delete [] Color;}
   virtual void print()=0;
};
```

```
class AutoCompacto : public Auto {
public:
   AutoCompacto(float P, char* C) : Auto(P, C)
{}
   AutoCompacto() {}
   ~AutoCompacto() {delete [] Color;}
   void print() {
       cout << "Auto compacto: Precio $"
            << Precio << " Color " << Color
            << endl;
   }
};
```

```
class AutoDeportivo : public Auto {
public:
   AutoDeportivo(float P, char* C) : Auto(P, C)
{}
   AutoDeportivo() {}
   ~AutoDeportivo() {delete [] Color;}
   void print() {
       cout << "Auto deportivo: Precio $"
            << Precio << " Color " << Color
            << endl;
   }
};
```

```
main() {
   char* colores[] = {"Rojo", "Negro", "Gris", "Azul", "Verde"};
   Auto* dealer[10];
   for(int i=1; i<=10; i++) {
      if(i%2)
         dealer[i]=new AutoCompacto(i*1000.0,colores[i%5]);
      else
         dealer[i]=new AutoDeportivo(i*20000.0, colores[i%5]);
      dealer[i]->print();
   }
   //dynamic memory deallocation code goes here...
}
```

Output:

Nombre: _____ **Sección:** _____

Problema 3. (30 puntos) Procedural Abstraction

Las cuatro partes de este problema trabajan con versiones cada vez mas generales de una función que procesa los elementos de un arreglo de números y devuelve un resultado numérico. La función aplica otra función matemática a cada celda del arreglo para obtener un valor numérico. Luego acumula los resultados numéricos de cada celda obteniendo un resultado final.

Considera la siguiente función:

```
float f (float a[], int size)
{
    float result = 0;
    for (int I=0; I<size; I++) {
        result += a[I] * a[I] * a[I];
    }
    return result;
}
```

En este caso la función f aplica la funcion x^3 a cada celda y acumula la suma de los cubos.

a. (10 puntos) La función f podría generalizarse abstrayendo la operación que se aplica a cada celda del arreglo en un parámetro funcional que podíamos llamar **map**. Completa la definición de la nueva función que acumula la suma de los resultados de aplicar la función map a cada celda.

```
float f2 (float [] a, int size, float map(float x))
{









}
```

Nombre: _____ **Sección:** _____

b. (10 puntos) Podríamos generalizar aún mas abstrayendo la función que se utiliza para acumular los resultados. La nueva función podría no solo acumular sumas sino productos u otras operaciones aritméticas. Completa la definición de una función f3 que tenga un parámetro funcional adicional a los de f2 llamado **fold**. Fold debe ser una función que reciba dos floats y devuelva un float.

```
float f3 (float [] a, int size, float map(float x), float fold(float a, float b)) {




}
```

c. (10 puntos) Finalmente podríamos generalizar f3 para que pueda procesar arreglos de cualquier tipo de objecto utilizando templates. Provee la definición de f4 con sus correspondientes parámetros.

```
template <class TYPE>
float f4 (                                                    ) {




}
```

Nombre: _____ **Sección:** _____

Problema 4. (15 puntos) Criptografía

A. (10 puntos) El staff de ICOM4015 utilizan sus propios metodos de codificación para comunicarse los problemas del examen. Utilizando sus propias funciones de criptografía, Encript y Decript, cada uno tiene la implementacion de las formulas, pero para cada examen tienen una clave diferente "key".

```
int* Encript(char *s, int key){

        int size=strlen(s);
        int *result=new int[size];
        for(int i=0;i<size;i++){
                result[i]=(int)s[i]+key;
        }
        return result;
}
```

Implemente la funcion Decript para invertir el efecto de la función Encript:

```
char* Decript(int *s, int key){




}
```

B. (5 puntos) Menciona las tres principios o conceptos mas importantes sobre computación y programación que aprendiste en ICOM 4015:

1.

2.

3.