Objetivos

- ✓ Definir las características de la solución de un problema bajo el paradigma de la orientación a objetos.
- ✓ Mejorar la capacidad individual de comprender y resolver problemas que admitan una solución mediante programación.
- ✓ Representar una solución software usando UML y codificarla en lenguaje C++.
- ✓ Realizar POO aplicando buenas prácticas de Ingeniería de Software.

Metodología general

- ✓ Los temas fundamentales son tratados en clase y acompañados de apuntes.
- ✓ La resolución de los ejercicios se realiza parcialmente por cada alumno en horarios dispuesto por él y parcialmente en las horas de clase en laboratorio con el profesor.
- ✓ Es conveniente que el desarrollo de cada ejercicio de esta guía se realice de manera individual, aunque no hay impedimentos en trabajarlos en grupos reducidos de alumnos.
- ✓ La finalidad de los ejercicios no obligatorios es la de mejorar las habilidades individuales de programación para resolver cada ejercicio identificado como obligatorio al mismo tiempo que proveen ciertas definiciones o implementaciones que puedan ser reutilizadas en la solución de los ejercicios obligatorios.
- ✓ Eventualmente cada alumno puede ser designado para presentar, explicar y defender su solución ante el curso.
- ✓ En cada práctico se indican las herramientas generales a usar, pero es el alumno quien debe analizar, discutir y seleccionar las más apropiadas en la resolución de cada ejercicio en particular. Un caso especial es el del software de diseño/programación usado que resulta fijado por el profesor (para otros productos consultar).
- ✓ Existen un conjunto de ejercicios y actividades complementarias (no obligatorias) a esta guía, indicadas en el material de las clases teóricas. Algunas de ellas a realizar bajo EAD.

Aprobación

- ✓ Los trabajos prácticos se consideran aprobados cuando cumpla los siguiente requisitos:
  - o Cada ejercicio presenta una solución individual del problema, esto implica que no deben aparecer soluciones “gemelas” entre los integrantes del curso, salvo que previamente hayan manifestado resolverlo de manera grupal, en cuyo caso se presentará/evaluará sólo una solución del grupo.
  - o Cada ejercicio ha sido ejecutado sin errores por el profesor luego de recibir la implementación o en el laboratorio en presencia del alumno.
  - o Haber realizado (presentado) el informe del conjunto de ejercicios indicados como obligatorios. El formato y contenido de este informe se indica en clase.
  - o La solución y el informe finales se han realizado y entregado por correo (o en aula virtual) al profesor dentro de los plazos indicados en clase.

Plazos

- ✓ Cada Trabajo práctico inicia luego de que han sido tratados los conceptos teórico-prácticos correspondientes.
- ✓ La implementación y el informe de cada trabajo práctico pueden ser presentados hasta la fecha límite indicada en clase. Se aconseja realizar la entrega de manera temprana.

Bibliografía Fundamental

- ✓ DEITEL H. M. Y DEITEL P. J. (2009): Cómo Programar en C/C++ (6ª edición o anteriores), Prentice-Hall
- ✓ STROUSTRUP, B. (2009): Programming. Principles and Practice Using C++, Pearson Education
- ✓ CEBALLOS SIERRA, F.J.(2007): C/C++ Curso de Programación (3ª edición o anterior) , Ra-Ma
- ✓ ACERA GARCIA, M.A. (2010): C/C++, Anaya Multimedia
- ✓ BRONSON, G. (2007): C++ para Ingeniería y Ciencias. 2da edición. Cengage Learning Editores S.A., México.
- ✓ Apuntes de cátedra y referencias indicadas en clase

I. TRABAJO PRACTICO: Introducción a C++

A. Herramientas necesarias para resolver los ejercicios

1. Proyecto C++. Módulos. Lenguaje C++. Tipos de datos. Variables. Casting. Punteros. Operadores. Arreglos. Estructuras de control estándares. Funciones y procedimientos. Paso por valor y referencia. Entrada/salida estándar. Funciones de biblioteca. Diseño de algoritmos. Diseño de estructuras de datos. IDE. Técnicas de depuración: breaking, debugging, stepping, tracing.

2. Discusión realizada en clase acerca de las diferencias existentes entre C y C++

B. Consignas

1. Generar los N primeros términos de la serie dada. Mostrar la serie y la sumatoria.

✓ En la solución definir como mínimo 2 funciones denominadas: obtener_termino, calcular_suma.

$$\sum_{k=1}^{n} \frac{\cos^2\left(\frac{\pi}{2}-(k+1)\right)+\cos 2k}{2^k}$$

Donde n es dado por el operador.

2. Dada una matriz cuadrada de orden N (siendo 3 ≤ N ≤ 10) con valores enteros al azar entre 1 y Max, dado por el operador, obtener 2 vectores conteniendo los valores correspondientes a la diagonal principal y a la diagonal secundaria, respectivamente.

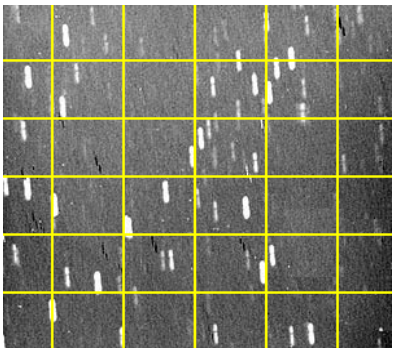
3. Implementar un programa capaz de generar una matriz de orden N x M (siendo 5 ≤ N, M ≤ 14) con valores float generados al azar, en el rango de 1 a 100, y presentar al operador un menú con los siguientes servicios:

a. Realizar el recorrido de la matriz desde la primera celda siguiendo una secuencia en espiral horaria, hasta la última celda ubicada en el "centro".

b. Listar los valores que no posean componente decimal y la cantidad de ellos.

c. Hallar los valores de recorrido, la media aritmética, desviación media y la desviación estándar para el conjunto de datos almacenado en la matriz

4. [opcional] Un observatorio astronómico requiere de un programa que analice una fotografía del cielo tomada por la noche.



La información de la fotografía está codificada de modo que cada elemento representa la cantidad de luz que se registró para cada zona durante la exposición.

Los valores registrados por el fotómetro van del 0 al 20.

La persona encargada de analizar la información supone que hay una estrella en el sector (i, j) si:

- el punto no se encuentra en las orillas de la fotografía (primera o última fila o columna)
- (A[i, j] + A[i -1, j] + A[i +1, j] + A[i, j -1] + A[i, j +1]) > 30

Se espera como resultado del análisis, una tabla B con un * en las parejas (i, y) en las que se supone que hay una estrella. El resto de la tabla debe quedar lleno de espacios.

Un dato adicional es que cada sector está identificado por una letra (como columna en el rango [A-Z]) y un número (como fila en el rango [1-9]). La cantidad puede variar según el caso.

Se desea que cada carga de datos por parte del operador sea almacenada en un archivo plano con forma de tabla (similar a lo observado en la tabla A del ejemplo dado).

✓ En la solución definir como mínimo 4 funciones: leer las dimensiones del sector, leer N valores de magnitud hasta que el operador decida terminar o se acabe el fotograma, analizar el fotograma, mostrar una tabla.

✓ Al producir la salida visualizar la matriz del fotograma y la matriz del análisis (de manera similar a la del ejemplo dado). Resulta indiferente que las matrices se muestren a la par o una debajo de otra, mientras puedan identificarse fácilmente tanto los valores como sus filas y columnas.

Por ejemplo:

Dada la tabla A de 6x8 (a la izquierda) se obtendría la tabla B de 6x8 (a la derecha)

0	3	4	0	0	0	6	8
5	13	6	0	0	0	2	3
2	6	2	7	3	0	10	0
0	0	4	15	4	1	8	0
0	0	7	12	6	9	10	4
5	0	6	10	6	4	8	0

	A	B	C	D	E	F	G	H
1								
2		*						
3								
4				*				
5				*	*		*	
6								