UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ESTRUCTURAS DE DATOS Y ALGORITMOS 1

M. en I. Fco. Javier Rodríguez García

Proyectos en Ingeniería

Alumno:	
Número de cuenta:	
Fecha:	
Calificación:	

xxx

1 Introducción al Aprendizaje Significativo y al Aprendizaje Orientado a Proyectos

1.1 Fundamentos

1.1.1. Aprendizaje Significativo, AS

Según la Wikipedia:

"El aprendizaje significativo es, según el teórico norteamericano David Ausubel, el tipo de aprendizaje en que un estudiante relaciona la información nueva con la que ya posee, reajustando y reconstruyendo ambas informaciones en este proceso. Dicho de otro modo, la estructura de los conocimientos previos condiciona los nuevos conocimientos y experiencias, y éstos, a su vez, modifican y reestructuran aquellos. Este concepto y teoría están enmarcados en el marco de la psicología constructivista.

El aprendizaje significativo ocurre cuando una nueva información se conecta con un concepto relevante preexistente en la estructura cognitiva, esto implica que las nuevas ideas, conceptos y proposiciones pueden ser aprendidos significativamente en la medida en que otras ideas, conceptos o proposiciones relevantes estén adecuadamente claras y disponibles en la estructura cognitiva del individuo y que funcionen como un punto de anclaje a las primeras.

Es decir: en conclusión, el aprendizaje significativo se basa en los conocimientos previos que tiene el individuo más los conocimientos nuevos que va adquiriendo. Estos dos al relacionarse, forman una conexión y es así como se forma el nuevo aprendizaje, es decir, el aprendizaje significativo."

1.1.2. Aprendizaje Orientado a Proyectos, AOP

Según la Wikipedia:

"El objetivo de la técnica didáctica aprendizaje basado en proyectos (AOP), […], es que el alumno aprenda haciendo a través de un proyecto en el que trabaja en forma similar a sus futuras prácticas profesionales: poniendo en juego sus conocimientos, capacidades y habilidades individuales en un trabajo de equipo.

AOP es una estrategia de aprendizaje en la que los alumnos llevan a cabo actividades que se concretan en un producto que puede ser una herramienta, un servicio, un plan de negocios, una investigación, etc. en donde se aplica el aprendizaje de una materia o disciplina, o de un conjunto de éstas. Para la realización del proyecto se cuenta con un periodo determinado que corresponde a la duración del curso en el que se realiza."

1.2 Actividades

1.2.1.

1.2.2.

2 Proyectos en ingeniería

"El científico explora lo que existe,
y el ingeniero crea lo que nunca ha existido"
– Dr. Theodore von Karman –

2.1 Introducción

Cualquier proyecto de ingeniería debe, sistemáticamente, ser descompuesto en diferentes etapas, cada una con un propósito específico, con el objetivo final de llevar a buen término dichos proyectos.

La metodología presentada a continuación ha demostrado, con la experiencia, dar buenos resultados.

2.2 Metodología

La metodología descrita a continuación está basada en etapas que son ejecutadas dentro de un patrón iterativo, es decir, una vez que una etapa está lista, entonces se pasa a la siguiente, pero si en la nueva etapa se encontraran incongruencias, entonces es posible regresarse a la etapa anterior para subsanar todo aquello que sea necesario, y toda esta ida y vuelta entre etapas se repite cuantas veces se requiera. (Esta metodología es diametralmente opuesta a la conocida como "Diseño en cascada", donde no es posible volver a la etapa anterior.)

No se habla de ningún campo de la ingeniería en particular, sino que esta metodología puede aplicarse a cualquier campo de ésta. Por supuesto que si particularizamos a una rama de la ingeniería en específico (por ejemplo, Ing. en Computación o en Ing. Electrónica) podremos agregar o modificar las etapas referidas; o dentro de cada etapa, asignar tareas que son muy específicas de estas ramas. Por ejemplo, no es lo mismo tener que construir un modelo de un puente para probar en una mesa vibratoria que que construir un circuito electrónico que será expuesto a temperaturas extremas, o que probar un software de seguridad con ataques simulados de hackers. Sin embargo, el común denominador (cada etapa) frecuentemente prevalece en los proyectos de ingeniería.

2.3 Etapas

Especificaciones <-> **Análisis** <-> **Diseño** <-> **Desarrollo** <-> **Pruebas** <-> **Producto final**

El símbolo "<->" representa una flecha con dos terminaciones, lo que significa que se puede ir de la etapa a la izquierda a la que está a la derecha y viceversa.

Es ampliamente recomendable no saltarse etapas, lo cual sucede con mucha frecuencia en proyectos con mala planeación o que han sido iniciados con premuras de tiempo.

El alumno encontrará en la siguiente tabla cada etapa con una breve lista de las tareas que se deben llevar a cabo en cada una de ellas.

Etapa	*Tareas*
ESPECIFICACIONES	• Es lo que el cliente quiere • Es un producto original o sobre un modelo existente • Seguimiento de normas • Costos • Tiempo estimado
ANÁLISIS	• Viabilidad técnica • Viabilidad económica • Costos reales (tiempo y dinero)
DISEÑO	• Búsqueda de soluciones • Análisis de soluciones • Descomposición • Organización • Obtención de tareas • Asignación de tareas y tiempos • Planteamiento de modelos o prototipos
DESARROLLO	• Aplicación de técnicas, conocimientos, experiencia y sentido común • Realización de los modelos o prototipos • Aquí se lleva a cabo la verdadera realización del producto
PRUEBAS	• Pruebas estáticas y dinámicas • Pruebas de estrés • Pruebas de conformidad a normas • Pruebas de conformidad a las especificaciones originales

3 Metodología

3.1 Elección del tema

3.1.1. Búsqueda de problemas

Para que al alumno se le facilite la labor de identificar un problema puede seguir una de las siguientes dos metodologías.

1. **Búsqueda de proyectos sociales**. Por el carácter social de la UNAM, y porque los recursos son limitados tanto económicos, como de tiempo e infraestructura, el alumno deberá identificar diversos problemas o necesidades reales en su entorno o comunidad, o situaciones en las que sus conocimientos de ingeniería pudieran mejorar una situación o sistema (es decir, de algo que ya existe, buscar la forma de hacerlo más eficiente).
2. **Búsqueda de proyectos comerciales**. El alumno identificará problemas de carácter comercial que generen un bienestar a la sociedad (o a un segmento en particular de ésta) a través de una remuneración económica por su trabajo. Esta remuneración puede provenir de vender directamente su producto, o por un un pago hecho por un empresario.

Para cualquier de los casos que haya escogido podría conducir una lluvia de ideas y generar un mapa mental de los diversos problemas que le gustaría abordar.

3.1.2. Búsqueda de soluciones

De los varios problemas o situaciones identificados en la actividad anterior el alumno deberá buscar, analizar y presentar soluciones viables que puedan ser resueltas en un tiempo razonable y con recursos reales de tiempo, dinero, conocimientos actuales, dificultad de implementación y de aceptación social, etc.

Es importante señalar que debido a que la materia es sobre estructuras de datos y algoritmos, tanto los problemas identificados como las posibles soluciones deberán estar enfocadas hacia esta rama de la ingeniería (vea “Aspectos a calificar”, en las bases del concurso).

Puede conducir una lluvia de ideas y generar un mapa mental de las diversas soluciones que podría dar a su problema.

TABLAProblema Solución

3.2 Justificación

En la vida real los recursos económicos y de infraestructura son escasos, ya sea que provengan de instituciones de carácter social o de grandes corporaciones, por lo que éstos se destinan únicamente a problemas cuya solución sea real, viable y que dejen algún tipo de beneficio (social o económico, dependiendo de la fuente de financiamiento). En otras palabras, los proyectos personales son raramente patrocinados. Es así que cuando el alumno comience a desarrollar la justificación de su propuesta de solución deberá visualizarse frente a un comité que lo evaluará para decidir si se le otorgan los recursos o no a su propuesta. (Personalmente me agrada la serie de la cadena BBC de Londres "Dragon's Den". El alumno puede ver algunos capítulos en en canal YouTube para que se forme una idea del proceso real de pararse frente a un grupo de inversionistas y presentar su producto o propuesta esperando recibir recursos. El alumno puede realizar la siguiente búsqueda "youtube dragon's den español subtitulado" en el buscador web de su preferencia .)

Una forma de llevar a cabo la justificación es respondiendo a las preguntas dadas a continuación. Tenga en cuenta que deberá responderlas en forma de prosa, no de respuesta a una pregunta, y que su escrito deberá tener una estructura de informe o resumen, no de respuestas a un cuestionario (las preguntas no tienen un orden particular):

- ¿Qué va a hacer?
- ¿Qué quiere hacer?
- ¿Porqué lo quiere hacer?
- ¿En qué se va a beneficiar la comunidad?
- ¿Cuánto va a costar?
- ¿Cuánto tiempo va a tomar?

- ¿Cuántos y cuáles recursos va a necesitar?
- ¿Habrán ganancias comerciales? ¿De cuánto van a ser?
- Durante el desarrollo y a la culminación del proyecto ¿Qué va a aprender el alumno de todo esto?

3.3 Hipótesis o Plan de Trabajo

Según la Wikipedia:

"Una hipótesis científica es una proposición aceptable que ha sido formulada a través de la recolección de información y datos, aunque no esté confirmada, sirve para responder de forma alternativa a un problema con base científica."

Con respecto a su proyecto es la propuesta de solución que Ud. realiza antes de poner manos a la obra. También se puede considerar como el plan general a seguir para lograr su objetivo, y por supuesto, listar los objetivos que se habrán de alcanzar al final del mismo.

Una solución de ingeniería involucra una metodología similar a una hipótesis, de ahí que ésta se haya tomado como título principal de este entregable. La definición dada de hipótesis menciona "... la recolección de información y datos ..."; en terminos de un proyecto de ingeniería esto significa "la descripción clara, concisa y sin ambigüedades de lo que el cliene quiere". En la ingeniería de software se le conoce formalmente como los requerimientos del cliente.

En la práctica el cliente raramente sabe en forma concisa y sin ambigüedades lo que quiere. Desafortunadamente, cuando no se tienen objetivos claros los proyectos no llegan a un final feliz. Es por ello que Ud. debe definir en forma clara lo que quiere, y a partir de ello, buscar las mejores metodologías y herramientas para lograrlo.

Suponiendo que los requerimientos sean claros, es importante mencionar que si la hipótesis es errónea, entonces los objetivos no se van a lograr porque a la mitad del proyecto la hipótesis deberá ser modificada, trayendo en consecuencia: proyectos que se extienden mucho más allá

del tiempo estimado (si es que se logran terminar), proyectos que superan el presupuesto original, atrasos con terceros que dependen del proyecto, que el cliente le diga "esto no fue lo que pedí", consecuencias legales, etc. Es por ello que su hipótesis o plan de trabajo deberá ser desarrollado con mucho cuidado.

3.3.1. Vista de usuario

Una forma de definir los requerimientos del sistema a desarrollar es poniéndose en el lugar del usuario: ¿quién usa al sistema? ¿cómo es el perfil del usuario promedio del sistema? ¿cómo se usa? ¿qué datos se le deben proporcionar al sistema? ¿qué resultados entregará el sistema?. En clase vimos algo que le llamamos la vista del usuario, lo cual era una descripción de lo que el sistema tenía que hacer. (Ver el ejercicio 3 del primer parcial).

Actividad

Describa a su sistema desde el punto de vista del usuario. En este punto Ud. no tiene que mencionar si va a usar arreglos o listas enlazadas, ni nada de eso; simplemente debe explicar de la mejor forma posible la forma en que el sistema funciona. Puede utilizar, además de texto, imágenes, dibujos, bocetos, fotografías, etc, y todo aquello que Ud. crea que ayudará a la mejor descripción de su sistema.

Nota: Los requerimientos completos y perfectos no existen, así que no pierda mucho tiempo tratando de alcanzar lo inalcanzable. Solamente la experiencia y el sentido común le dirán si los requerimientos han sido insuficientes, o si ya se excedieron las expectativas y nada más se está perdiendo el tiempo. Quizás le interese ver esta definición del ciclo de un proyecto de ingeniería.

3.3.2. Vista lógica o abstracta

Una vez que los requerimientos están bien definidos es tiempo de comenzar a ver las estructuras de datos de alto nivel (abstractas) y los algoritmos que la aplicación ha de implementar.

Puede notar que mi mención a las estructuras de datos y algoritmos está en plural. Una aplicación no trivial usa dos o más de cada uno. Para determinar qué estructuras de datos o algoritmos tiene que implementar debe volver a los requerimientos y ver la forma en que los

datos serán introducidos (por teclado, por archivo, o ambas, por internet), y la forma en que serán presentados (pantalla, archivo, o ambas, por internet). También debe poner atención a si los datos serán procesados (ordenamiento y búsqueda).

Actividad

Describa las diferentes estructuras de datos, los diferentes algoritmos y las diferentes formas en que

todos estos van a interactuar. Es decir, puede tener una lista doblemente enlazada donde un campo de

datos puede ser una lista simplemente enlazada. A su vez, tanto la lista doblemente enlazada como la

simplemente enlazada quizás necesiten ser ordenadas, y por supuesto, realizar búsquedas en ambos

tipos de listas.

Nota: Aunque sabemos que la implementación de su proyecto será en C, en este momento no hemos hablado absolutamente nada sobre código. Todo lo que hemos hecho hasta este punto es abstracto. Dicho de otra forma, hemos estado escribiendo nuestro plan.

3.4 Desarrollo de software

Aquí puede encontrar una explicación corta sobre el desarrollo de software. Además, le recomiendo dos herramientas que le serán muy útiles:

Control de versiones. Puede usar Subversion o Git. Si usa Git puede crear sus repositorios en:

GitHub. Los repos gratuitos deben ser públicos =(

BitBucket. Los repos gratuitos pueden ser privados =) (es el que yo uso)

Un planificador como Planner.

3.5 Pruebas

3.5.1. Depuración con GDB

Se usa para depurar (buscar errores) después de que el código fue escrito.

3.5.2. TDD: Test Driven Development

Esta metodología de desarrollo alienta la escritura de pruebas mientras se está escribiendo el código.

3.5.3. Pruebas de regresión

Cuando hacemos cambios a un software, ya sea por corrección de errores, o porque agregamos funcionalidades nuevas, debemos asegurarnos que todo el sistema sigue funcionando; esto es, que dichos cambios no lo han afectado en forma negativa. Aquí se tiene una explicación más amplia.

3.6 Desarrollo

3.7 Prototipo

4 Bases

4.1 De los participantes

Este es un concurso interno con fines de alentar la competitividad y conocimientos del alumno en la materia y materias relacionadas.

La participación debe ser en equipo con el número de integrantes mostrado en la siguiente tabla. Y ni el número de integrantes del equipo, ni el nombre de alguno o algunos de ellos podrá ser cambiado una vez que el proyecto se registró.

Grupo	***Participantes por equipo***
06	Desde 2 hasta 3
13	Desde 1 hasta 2

Sólo está permitido un proyecto por equipo.

4.2 De los aspectos a evaluar

Que el proyecto funcione según el nombre, descripción y objetivos planteados. El proyecto puede ser de software o hardware (Raspberry Pi o similar, Arduino o similar, etc.)

- Que el proyecto tenga aplicación práctica.
- El uso de estructuras de datos.
- Programación en el lenguaje C99 o posterior.
- El uso de abstracciones a través de TDUs.
- Utilización de algunas herramientas para desarrollo de software, por ejemplo, el desarrollo conducido por pruebas (TDD, *Test Driven Development*).

4.3 Del jurado

El jurado soy yo.

El jurado se reserva el derecho de descalificar a cualquier equipo que a su consideración hubiera recibido ayuda externa para la realización del proyecto. El jurado decidirá la o las formas para probar que fueron los alumnos quien lo llevaron a cabo.

4.3.1. Restricciones

- La calificación máxima del alumno en el semestre es de 10.
- Los puntos sobrantes (si los hubiera) son instransferibles.
- Por cada alumno del equipo: Si éste no alcanza el 6.00 con los elementos convencionales de calificación (antes de aplicar la escala) o tiene reprobado el laboratorio, entonces el proyecto no le será tomado en cuenta.
- Ningún proyecto extemporáneo será tomado en cuenta.
- Una vez que el proyecto ha sido aceptado no está permitido cambiar el nombre, descripción ni objetivos de éste.
- Todos los proyectos serán de código abierto apegándose a las licencias GPL y Creative Commons vigentes a la fecha de recepción de los proyectos. Si el alumno no está de acuerdo, entonces deberá abstenerse de concursar.

4.4 De los entregables

- El primer paso es entregar este documento con los siguientes campos debidamente llenados: Carátula, Descripción, Objetivos, Metodologías y Plan.
- Cuando el jurado lo indique el alumno deberá entregar un avance del proyecto.
- Cuando el jurado lo indique el alumno deberá entregar este documento completamente llenado y el código fuente del proyecto terminado.
- El proyecto deberá compilar en Linux utilizando al compilador GCC 99 o posterior.

- La documentación del código fuente deberá estar en alguno (sólo uno) de los formatos que la aplicación Doxygen acepta.

4.5 Otros

El jurado decidirá y hará llegar con tiempo todos aquellos puntos no contemplados en estas bases.

5 Entregables

5.1 HORARIO DE ENTREGA

Para ambos grupos, el 03 y el 06:

La hora máxima de entrega será a las 12:00 horas (medio día) del sábado 3 de junio. Presten atención a los siguientes puntos.

5.2 ARCHIVOS

El ejercicio NO deberá tener carpetas. Todos sus archivos en el mismo lugar. En caso de que las hubieran utilizado, su instrucción de compilación deberá de contemplarlo.

Los archivos fuente deberán estar en un sólo archivo comprimido de nombre proyecto.zip (o .tar.bz, o .tar.bz2, etc, dependiendo del programa compresor que usen). La rúbrica deberá estar incluída en este archivo comprimido, como se detalla más adelante.

No olviden que para el proyecto deberá existir un archivo instrucciones.txt con la instrucción de compilación correspondiente, completa y correcta (ver los siguientes puntos).

SU PROYECTO DEBERÁ COMPILAR EN UNA TERMINAL DE LINUX. Esto es, los proyectos creados en alguna IDE (Codeblocks, Eclipse, Xcode, etc.) NO serán tomados en cuenta si no cumplen lo anterior. Si el proyecto no compila debido a dependencias no satisfechas (windows.h, conio.h, por ejemplo, o cualquier otra similar), tendrán 5 puntos menos en la calificación del mismo.

Si su proyecto contempla el uso de bibliotecas no estándar, éstas deberán ser incluídas en el

archivo comprimido, y por supuesto, su instrucción de compilación deberá contemplarlo. Recuerden que uso Linux Mint de 64 bits (LM es un derivado de Ubuntu).

5.3 FORMATO DEL CORREO ELECTRÓNICO

Al igual que la vez anterior su archivo comprimido deberán enviarlo a la dirección:

eda1.fiunam@yahoo.com.mx

Ningún proyecto que no llegue a este correo será aceptado.

En el asunto del correo escribirán:

Para el grupo 06:

```
G06-TF-NUMERO_DE_CUENTA_SIN_GUION_DEL_RESPONSABLE_DEL_EQUIPO
```

Para el grupo 13:

```
G13-TF-NUMERO_DE_CUENTA_SIN_GUION_DEL_RESPONSABLE_DEL_EQUIPO
```

Por ejemplo:

```
G06-TF-123456789
```

```
G13-TF-987654321
```

(TF significa "Trabajo final".)

En el cuerpo del correo deberán escribir su nombre, empezando por apellidos, seguido de su correo electrónico, Y NADA MÁS! El nombre en una línea y su correo en otra!!!!!!!!!! Por ejemplo:

```
Rodríguez García Fco. J.
```

```
eda1.fiunam@yahoo.com.mx
```

5.4 FORMATO DE LA ENTREGA EN PAPEL

Como ya no nos veremos en la escuela, sino hasta la segunda vuelta, no habrá entrega en

papel. Pero tomen en cuenta lo siguiente:

5.4.1. Rúbrica

Antes que cualquier otra cosa, LEÁNLA, y observen los valores y los sub-aspectos. Es muy similar a las anteriores, pero no es igual.

Anéxenla al pack de sus códigos fuente. El PDF de ésta acepta que escriban texto (como las carátulas de las prácticas). Van a llenarla con los datos de los integrantes del equipo. El primer registro es para el responsable del mismo, y será éste quien envíe el archivo comprimido, y con quien mantendré comunicación.

En caso de que no puedan escribir sobre el PDF, entonces podrán imprimirlo, llenarlo a mano, escanearlo, pegarlo en LibreOffice Writer o Word, generar un nuevo PDF, y luego agregarlo al pack comprimido (NO vayan a enviar el .odt o el .docx, necesito el PDF). A diferencia del último examen, eso no nos afecta para nada.

5.4.2. Pseudo-códigos

Si desean agregar sus pseudo-códigos pueden hacerlo, pero de la siguiente manera: Los escanean, pegan las imágenes en LibreOffice Writer o Word, generan un PDF de este documento y lo anexan al pack comprimido. El nombre del PDF será pseudo-códigos.pdf. !NO VAYAN A ENVIAR IMÁGENES SUELTAS! No los pasen en limpio, no los capturen en la computadora, no los hagan después del código. Existe una aplicación para el celular, entre muchas otras, CamScanner (CS), cuya versión gratuita les puede ayudar a escanear sus documentos en caso de que no tengan un escáner a la mano.

5.4.3. Información del proyecto

Van a escribir un documento formalizando la información que ya me habían entregado: Elección del tema, Justificación e Hipótesis. Lo mencionado ya está evaluado, solamente lo quiero por escrito en el formato (de texto) que ustedes elijan. Les menciono esto para que no vayan a perder tiempo agregando o modificando lo que ya tenían; usen su tiempo en lo que les falte del desarrollo de su proyecto. Este documento se va a llamar

información_del_proyecto.pdf.

5.5 ENTREGABLES

Van a entregar lo siguiente (todo en el mismo pack comprimido):

- Los códigos fuente (según la Sección 2).
- La rúbrica (según 4.1).
- Los pseudo-códigos, diagramas de flujo, diagramas de clases, etc. (según 4.2).
- La información del proyecto (según 4.3).

6 NOTAS EXTRAS

Y como siempre, no hay entregas extemporáneas ni físico ni electrónico.

7 CALENDARIO

Espero haber sido claro con las indicaciones, pero si algo se me pasó, por favor háganmelo saber para corregirlo o agregarlo.

6 Referencias y bibliografía recomendada

- [DEITELxx]
  Deitel, H. M., Deitel, P. J. **Cómo programar en C/C++**. 2da. ed. ESPAÑA: McGraw-Hill, 20xx.
- [JOYANES04]
- [JOYANES05]
- Referencia del lenguage C
  http://www.cplusplus.com/reference/cstring/