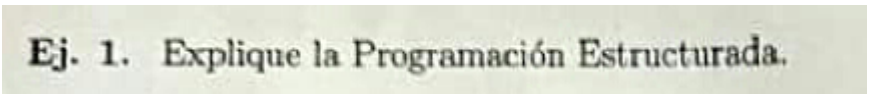
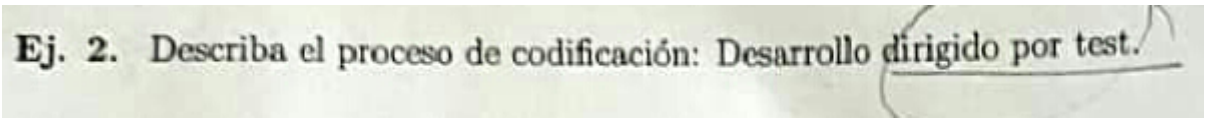
Pseudo Resumen INGSoft Parcial 2 con respuestas a examenes anteriores

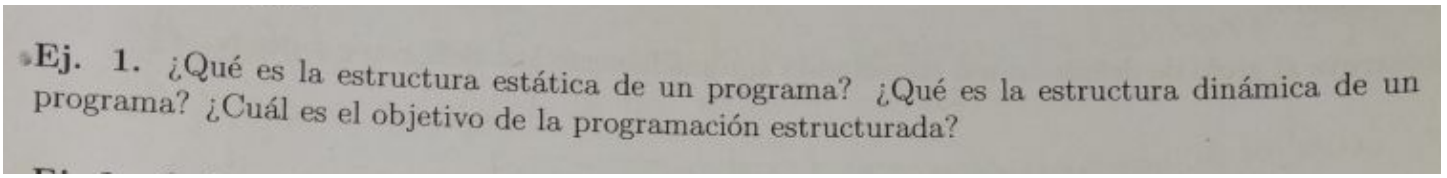
Codificación


Ej. 1. Explique la Programación Estructurada.

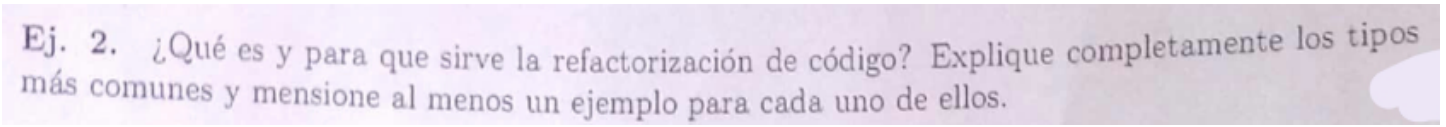
NOTA: 1 aparicion en examenes pasados


Ej. 2. Describa el proceso de codificación: Desarrollo dirigido por test.

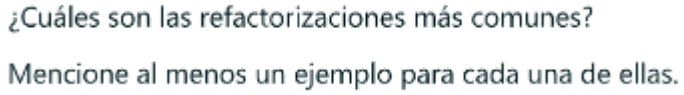
NOTA: 3 apariciones en examenes pasados


Ej. 1. ¿Qué es la estructura estática de un programa? ¿Qué es la estructura dinámica de un programa? ¿Cuál es el objetivo de la programación estructurada?

NOTA: 1 aparicion en examenes pasados

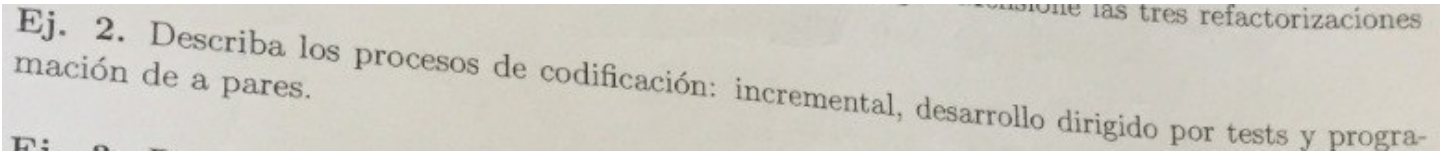

Ej. 2. ¿Qué es y para que sirve la refactorización de código? Explique completamente los tipos más comunes y mensione al menos un ejemplo para cada uno de ellos.

NOTA: 3 apariciones en examenes pasados


¿Cuáles son las refactorizaciones más comunes?

Mencione al menos un ejemplo para cada una de ellas.

NOTA: 2 apariciones en examenes pasados


Ej. 2. Describa los procesos de codificación: incremental, desarrollo dirigido por tests y programación de a pares.

NOTA: 2 apariciones en examenes pasados

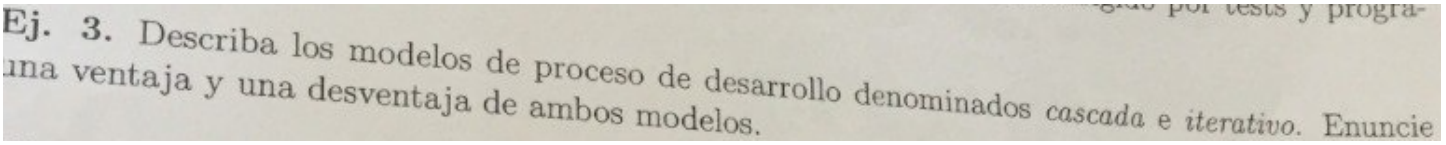
Modelos de proceso de desarrollo

Explique exhaustivamente el proceso de desarrollo: Iterativo.

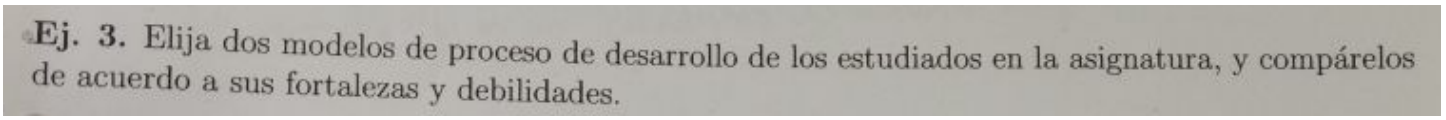
NOTA: 1 aparicion en examenes pasados

2) Explique exhaustivamente el proceso de desarrollo: Prototipado.

NOTA: 2 apariciones en examenes pasados

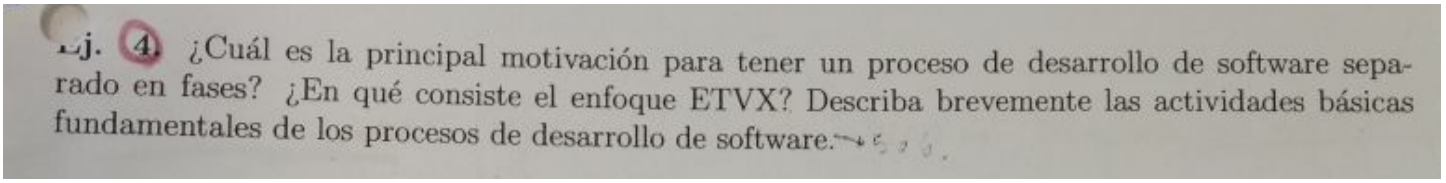

Ej. 3. Describa los modelos de proceso de desarrollo denominados *cascada* e *iterativo*. Enuncie una ventaja y una desventaja de ambos modelos.

NOTA: 1 aparicion en examenes pasados

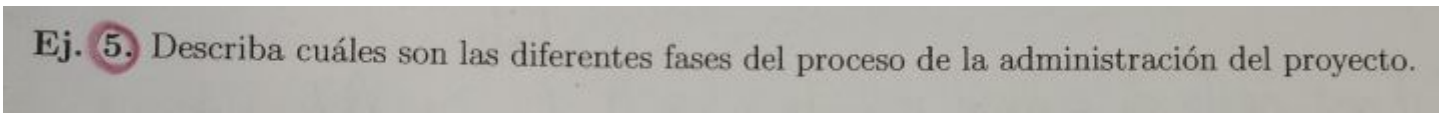

Ej. 3. Elija dos modelos de proceso de desarrollo de los estudiados en la asignatura, y compárelos de acuerdo a sus fortalezas y debilidades.

NOTA: 1 aparicion en examenes pasados

Procesos de desarrollo


Ej. 4. ¿Cuál es la principal motivación para tener un proceso de desarrollo de software separado en fases? ¿En qué consiste el enfoque ETVX? Describa brevemente las actividades básicas fundamentales de los procesos de desarrollo de software.

NOTA: 1 aparicion en examenes pasados


Ej. 5. Describa cuáles son las diferentes fases del proceso de la administración del proyecto.

NOTA: 1 aparicion en examenes pasados

Testing

Ej. 4. ¿Qué es y para qué sirve el oráculo de test? ¿Qué son y para qué sirven los criterios de selección de tests? ¿Cúal es la diferencia entre el testing de caja blanca y el testing de caja negra?

NOTA: 3 apariciones en examenes pasados con variantes sobre lo de la diferencia de caja blanca y negra

Ej. 4. Explique el testing de caja blanca de cobertura de ramificaciones. Dé un ejemplo en el que un error pueda ser encontrado por el criterio de cobertura de ramificaciones y no por el de cobertura de sentencia.

NOTA: 1 aparicion en examenes pasados

¿Qué es el testing de caja negra? Enuncie al menos dos criterios dentro de esta categoría.

NOTA: 3 apariciones en examenes pasados

4) Explique el criterio de cobertura de sentencia, dando un ejemplo no mencionado en el teórico.

NOTA: 2 apariciones en examenes pasados

Ej. 6. Explique y de un ejemplo del testing de caja negra de análisis de valores límites

NOTA 1 aparicion

Ej. 7. ¿Qué son y para qué sirven los criterios de selección de tests?

NOTA 1 aparicion

Ej. 8. Explique y de un ejemplo del testing de caja negra de particionado por clases de equivalencia.

NOTA 1 aparicion

Planeamiento

Ej. 3. Describa exhaustivamente el modelo COCOMO.

Ej. 4. Explique para qué sirve el modelo COCOMO. Describa sus pasos y enumere además al menos 2 atributos que influyen en su estimación final.

Ej. 9. Describa los pasos básicos asociados a la aplicación del modelo COCOMO para la estimación de esfuerzos. Enumere además al menos 2 atributos que influyen en la estimación final asociada a este modelo.

Ej. 5. Describa exhaustivamente la Administración de Riesgos.

PRÁCTICO

Parte práctica

Ej. 6. Considere el siguiente fragmento de código:

```
1. int ordenado, i, j, aux;
2. int main()
3. {
4.     printf("Lista original:\n");
5.     for(i=0;i<4;i++){
6.         (i<3)?printf("%d,",lista[i]):printf("%d.",lista[i]);
7.     }
8.
9.     for(i=0; i<4;i++){
10.         for(j=0;j<3;j++){
11.             if(lista[j]>lista[j+1]){
12.                 aux= lista[j];
13.                 lista[j]= lista[j+1];
14.                 lista[j+1]= aux;
15.             }
16.         }
17.     }
18.
19.     printf("\nLista ordenada:\n");
20.     for(i=0;i<4;i++){
21.         (i<3)?printf("%d,",lista[i]):printf("%d.",lista[i]);
22.     }
23.     return 0;
24. }
```

(a) Construya el grafo de definición-uso etiquetando apropiadamente los conjuntos: def., uso-c y uso-p. Usar como referencia los número de líneas dados.

(b) Describa el criterios de coberturas de todas las definiciones. Construya un conjunto de casos de test para este código que cumpla con dicho criterio para la variable: lista[0].

5) Considere el siguiente fragmento de código:

```
Function Kr(n, a: longint): extended;
01.  var tot: extended;
02.          i, j, k, l: longint;
03.          d: array [1..100] of longint;
04.  begin
05.  tot:= 0;
06.  quicksort(1, a);
07.     for i:= 1 to n do
08.        d[i]:= i;
09.     for i:= 1 to a do
10.        begin
11.        k:= d[e[i].v1]; l:= d[e[i].v2];
12.        if (k<>1) then
13.           begin
14.           tot:= tot + e[l].k;
15.           for j:=1 to n do
16.              if (d[j]=1) then d[j]:= k;
17.           end;
18.        Kr:= tot;
19.        end;
```

a) Construya un conjunto de casos de tests para este código que cumpla con el criterios de cobertura de ramificaciones.

b) Construya el grafo de definición-uso etiquetando apropiadamente los: def., uso-c y uso-p.

c) ¿Porqué las preguntas a y b están en el orden dado?

6) Describa exhaustivamente los pasos asociados a la aplicación del modelo COCOMO para la estimación de esfuerzos. Enumere además al menos 3 atributos que influyen en la estimación final asociada a este modelo.

Considere el siguiente fragmento de código:

```
public static bool Contiene(string palabra, string cadena)
01. {
02.     for (int i=0; i<=palabra.Length-cadena.Length; i++)
03.         if (palabra[i]==cadena[0])
05.         {
06.             bool contenida = true;
07.             for (int j=1; j<cadena.Length; j++)
08.             {
09.                 if (cadena[i+j] != cadena[j])
10.                     contenida = false;
11.                 if (contenida)
12.                     return true;
13.             }

    .       return false;
16. }
```

a) Construya un conjunto de casos de tests para este código que cumpla con el criterios de cobertura de ramificaciones.

b) Construya el grafo de definición-uso etiquetando apropiadamente los: def., uso-c y uso-p.

c) ¿Porqué las preguntas a y b están en el orden dado?

Ej. 7. Considere el siguiente fragmento de código:

```
int ordenado, i, j, aux;
int main()
{
   printf("Lista original:\n");
   for(i=0;i<4;i++){
       (i<3)?printf("%d,",lista[i]):printf("%d.",lista[i]);
   }
    for(i=0; i<4;i++){
       for(j=0;j<3;j++){
           if(lista[j]>lista[j+1]){
               aux= lista[j];
               lista[j]= lista[j+1];
               lista[j+1]= aux;
           }
       }
   }
   printf("\nLista ordenada:\n");
   for(i=0;i<4;i++){
       (i<3)?printf("%d,",lista[i]):printf("%d.",lista[i]);
   }
   return 0;
}
```

(a) Construya conjuntos de casos de test para este código que cumplan con los criterios de cobertura de ramificaciones.

(b) Construya el grafo de definición-uso etiquetando apropiadamente los: def., uso-c y uso-p.