Nombre: Juan Camilo Parra Sanchez

ID: 917079

Hoja de Trabajo N.º 2: Un Estudiante

Enunciado. Analice el siguiente enunciado e identifique el mundo del problema, lo que se quiere que haga el programa y las restricciones para desarrollarlo.

Se desea construir una aplicación para el manejo de información de los cursos que está tomando un estudiante. El estudiante toma solo 4 cursos en el semestre. Los datos personales del estudiante que maneja la aplicación son código, nombre y apellido.

De cada curso se conoce:

- Código. Es el identificador del curso y no puede haber dos cursos con el mismo código.
- Nombre.
- Departamento. Puede ser Matemáticas, Física, Sistemas o Biología.
- Cantidad de créditos.
- Nota obtenida en el curso. Este valor debe estar entre 1.5 y 5.

Para poder calcular el promedio del estudiante, se deben ponderar las notas, teniendo en cuenta la cantidad de créditos de las materias. Para esto, para cada curso se debe multiplicar la nota del curso con su cantidad de créditos, sumar estos valores y dividir esta suma por la cantidad total de créditos vistos por el estudiante. Por ejemplo, si el estudiante ha terminado dos materias, “Cálculo 1” y “Física 1”, la primera de 4 créditos y la segunda de tres, con las siguientes notas:

- Cálculo 1: 4,5
- Física 1: 3,5

El promedio del estudiante es:

- (4,5 * 4 + 3,5 * 3) / 7 = 4,07

Adicionalmente, se quiere poder saber si un estudiante está en prueba académica o si es candidato para beca. Para esto se debe tener en cuenta las siguientes reglas:

- Se considera que un estudiante está en prueba académica si su promedio es inferior a 3.25.
- Se considera que un estudiante es candidato a beca si su promedio es igual o superior a 4.75.

La aplicación debe permitir: (1) visualizar la información del estudiante, (2) visualizar la información de los cursos, (3) modificar la información de un curso, (4) asignar una nota a un curso (5) calcular el promedio del estudiante (6) indicar si el estudiante está en prueba académica, (7) indicar si el estudiante es candidato a beca.

Requerimientos funcionales. Especifique los siete requerimientos funcionales descritos en el enunciado.

Requerimiento Funcional 1

Nombre:	R1 – Visualizar la información del estudiante.
Resumen:	**Muestra la información personal del estudiante**
Entradas:	**Ninguna**
Resultado:	**Muestra la información del estudiante (codigo, nombre y apellido)**

Requerimiento Funcional 2

Nombre:	R2 – Visualizar la información de los cursos.
Resumen:	**Muestra la información de los cursos**
Entradas:	**Ninguna**
Resultado:	**Muestra la informacion de cada uno de los cursos**

Requerimiento Funcional 3

Nombre:	R3 – Modificar la información de un curso.
Resumen:	**Modifica la infrmacion de un curso en especifico**
Entradas:	**Código, nombre, departamento, créditos**
Resultado:	**La información del curso es actualizada**

Requerimiento Funcional 4

Nombre:	R4 - Asignar una nota a un curso.
Resumen:	**Permite asignar una nota a un curso en especifico**
Entradas:	**Codigo del curso y la nota a asignar**
Resultado:	**Se actualiza la nota del curso**

Requerimiento Funcional 5

Nombre:	R5 - Calcular el promedio del estudiante.
Resumen:	**Calcula el promedio de notas del estudiante, basado en las notas de este y los créditos del curso**
Entradas:	**Ninguna**
Resultado:	**El promedio del estudiante**

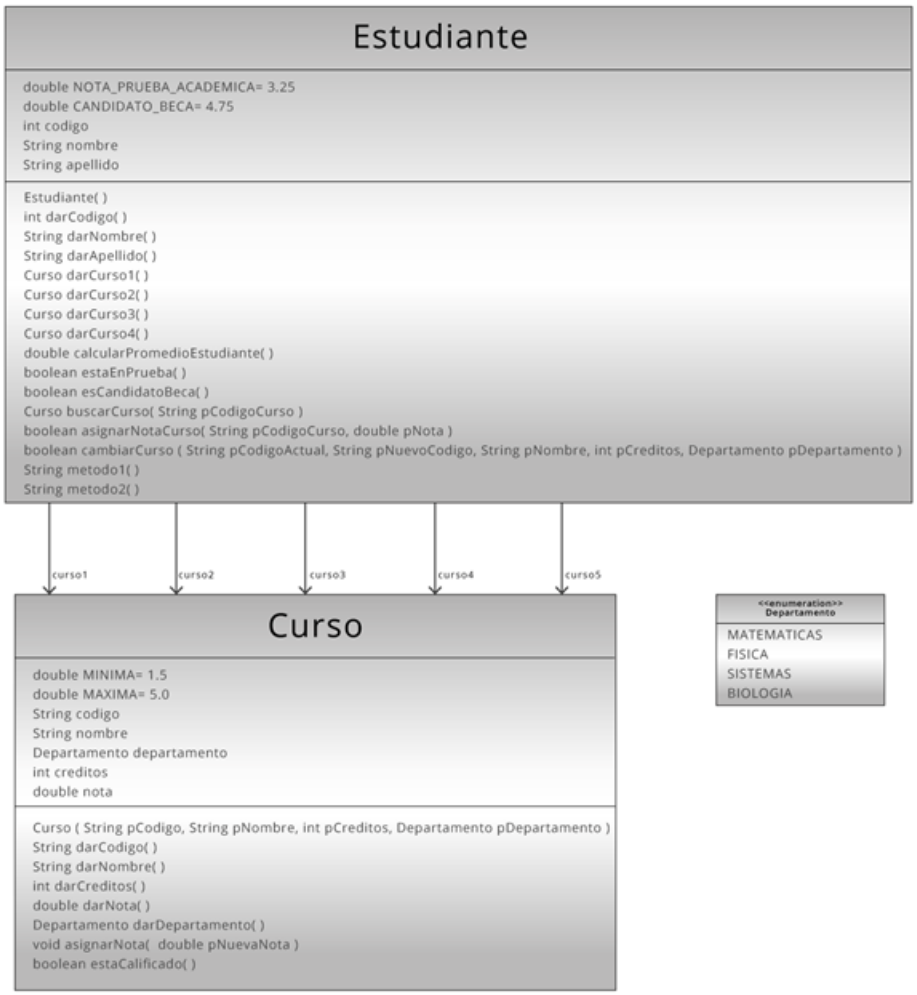
Requerimiento Funcional 6

Nombre:	R6 - Indicar si el estudiante está en prueba académica.
Resumen:	**Verifica si el promedio del estudiante es mayor a 3.25**
Entradas:	**Promedio del estudiante**
Resultado:	**Si está en prueba académica retorna TRUE, y si no se encuentra en esta FALSE**

Requerimiento Funcional 7

Nombre:	R7 - Indicar si el estudiante es candidato a beca.
Resumen:	**Verifica se el promedio del estudiante es mayor a 4.75**
Entradas:	**Promedio del estudiante**
Resultado:	**Si es candidato a beca retorna TRUE, si no es candidato a beca retorna FALSE**

Modelo conceptual. Estudie el siguiente modelo conceptual.



Double = Real
String = Cadena de caracteres
Int = Entero

Declaración de las clases. Complete las declaraciones de las siguientes clases.

```
class Estudiante():
    NOTA_PRUEBA_ACADEMICA = 3.25
    CANDIDATO_BECA = 4.75
    #-------------------------------------------------
    # Constructor
    #-------------------------------------------------
    __method__ = "Constructor"
    __parameter__ = "Ninguno"
    __returns__ = "Ninguna"
    __Description__ = "metodo constructor de la clase"
    def  __init__(self, codigo, nombre, apellido):
    #-------------------------------------------------
    # Atributos
    #-------------------------------------------------
        self.codigo = codigo
        self.nombre = nombre
        self.apellido = apellido
```

```
class Curso():
    MINIMA = 1.5
    MAXIMA = 5.0
    #-------------------------------------------------
    # Constructor
    #-------------------------------------------------
    __method__ = "Constructor"
    __parameter__ = "Ninguno"
    __returns__ = "Ninguna"
    __Description__ = "metodo constructor de la clase"
    def __init__(self, codigo, nombre, creditos, departamento, nota=0):
    #-------------------------------------------------
    # Atributos
    #-------------------------------------------------
        self.codigo = codigo
        self.nombre = nombre
        self.creditos = creditos
        self.departamento = departamento
        self.nota = nota
```

Creación de expresiones. Para cada uno de los siguientes enunciados, escriba la expresión que lo representa. Tenga en cuenta la clase dada para determinar los elementos disponibles.

Curso	¿El nombre del curso es “Cálculo 1”?	**Curso.nombre == “calculo 1”**
Curso	¿El curso ya tiene una nota asignada?	**Curso.notaAsignada()**
Curso	¿El curso tiene más de tres créditos?	**Curso.creditos > 3**
Curso	¿El curso fue aprobado?	**Curso.fueAprobado()**
Estudiante	¿El código del estudiante es “1234”?	**Estudiante.codigo == “1234”**
Estudiante	¿El primer curso tiene una nota asignada?	**Estudiante.curso1.notaAsignada()**
Estudiante	¿El segundo curso pertenece al departamento de matemáticas?	**Estudiante.curso2.departamento == “Matematicas”**
Estudiante	¿Cuál es el promedio del estudiante?	**Estudiante.calcularPronedioEstudiante()**

Desarrollo de métodos. Escriba el código de los métodos indicados.

Clase: Curso Retorna el código del curso.	`def darCodigo(self):` `  return self.codigo`
Clase: Curso Indica si el curso ya fue calificado (tiene una nota distinta de cero).	`def estaCalificado(self):` `  return self.nota != 0`
Clase: Estudiante Retorna el nombre del estudiante.	`def darNombre(self):` `  return self.nombre`
Clase: Estudiante Indica si el estudiante ya tiene los cuatro cursos pertenecen al mismo departamento.	`def pertenecenMismoDepartamento(self):` `  mismoDepartamento = self.curso1.darDepartamento()` `  if self.curso2.darDepartamento() != mismoDepartamento:` `  return False` `  if self.curso3.darDepartamento() != mismoDepartamento:` `  return False` `  if self.curso4.darDepartamento() != mismoDepartamento:` `  return False`

	`` else: return True ```

Clase: Estudiante Calcula el promedio de los cursos que ya tienen nota. Si ningún curso tiene nota asignada, retorna cero.	``` def calcularPromedioEstudiante(self): sumaTotal = 0.0 creditosTotales = 0.0 if self.curso1.estaCalificado(): sumaTotal += self.curso1.darNota() * self.curso1.darCreditos() creditosTotales += self.curso1.darCreditos() if self.curso2.estaCalificado(): sumaTotal += self.curso2.darNota() * self.curso2.darCreditos() creditosTotales += self.curso2.darCreditos() if self.curso3.estaCalificado(): sumaTotal += self.curso3.darNota() * self.curso3.darCreditos() creditosTotales += self.curso3.darCreditos() if self.curso4.estaCalificado(): sumaTotal += self.curso4.darNota() * self.curso4.darCreditos() creditosTotales += self.curso4.darCreditos() if creditosTotales == 0: return 0.0 return sumaTotal / creditosTotales ```

Clase: Estudiante Busca y retorna el curso que tiene el código que se recibe como parámetro. Si ningún curso tiene dicho código, el método retorna null.	`def buscarCurso(self, pCodigoCurso):` `  if self.curso1.darCodigo() == pCodigoCurso:` `  return self.curso1` `  if  self.curso2.darCodigo() == pCodigoCurso:` `  return self.curso2` `  if  self.curso3.darCodigo() == pCodigoCurso:` `  return self.curso3` `  if  self.curso4.darCodigo() == pCodigoCurso:` `  return self.curso4` `  return None`
Clase: Estudiante Indica si el estudiante se encuentra en prueba académica. Retorna verdadero si está en prueba académica, false de lo contrario.	`def estaEnPrueba(self):` `  return self.calcularPromedioEstudiante() < self.NOTA_PRUEBA_ACADEMICA`