Boletín de ejercicios sobre Streams

Para los siguientes ejercicios cree una aplicación Java en la que defina las siguientes clases:

- `Student`: Representa un alumno. Dispone de los campos `id` (identificador, long), `name` (nombre, cadena), `age` (edad, entera), `group` (grupo clase, cadena), `grant` (cantidad de beca, entero), `grades` (notas, lista de objetos `Grade`).
- `Grade`: Representa una nota. Dispone de los campos `subject` (nombre de la asignatura, cadena) y `mark` (nota en dicha asignatura, float).
- `Database`: Representa la base de datos con la que trabajaremos. Dispondrá de un campo privado correspondiente a la lista de estudiantes. Además tendrá un método público denominado `queryAllStudents()` que retornará la lista de alumnos.

Inicializa la base de datos con los siguientes datos de los alumnos (en este orden):

id	name	age	group	grant	grades
1	Germán Ginés	23	1º CFGS DAM	2000	[PROGR, 8], [LM, 3]
2	Baldomero	21	1º CFGS DAM	0	[PROGR, 5], [LM, 4]
3	Ana Guerra	17	1º CFGM SMR	4000	[PROGR, 8]

Realiza en Java los métodos necesarios para mostrar por pantalla el resultado de las siguientes consultas, usando *streams* (y *optionals* cuando sea necesario).

1.- showLegalAgeStudentCount()

Debe mostrar los alumnos que sean mayores de edad. La salida sería:

```
Número de alumnos mayores de edad: 2
```

2.- showStudentNamesOrderAlphabetically()

Debe mostrar los nombres de los alumnos ordenados alfabéticamente. La salida sería:

```
Nombres de alumnos (orden alfabético):
Ana Guerra
Baldomero
Germán Ginés
```

3.- showFistTwoStudentsNames()

Debe mostrar los nombres de los dos primeros alumnos. La salida sería:

```
Nombres de los primeros dos alumnos:
Germán Ginés
Baldomero
```

4.- showStudentsNamesExceptTheFirstOne()

Debe mostrar los nombres de todos los alumnos menos del primero, La salida sería:

```
Nombres de alumnos (excepto el primero):
Baldomero
Ana Guerra
```

5.- showStudentsNamesUntilFirstNotLegalAgeOne()

Debe mostrar los nombres de todos los alumnos hasta que encuentre uno menor de edad. Éste ya no lo mostrará. La salida sería:

```
Nombres de alumnos (hasta que encontramos uno menor de edad):
Germán Ginés
Baldomero
```

6.- showStudentsSinceFirstNotLegalAgeOne()

Debe mostrar los nombres de todos los alumnos desde que encontremos uno menor de edad. Éste si lo mostrará. La salida sería:

```
Nombres de alumnos (desde que encontramos uno menor de edad):
Ana Guerra
```

7.- showDifferentSubjectsOrderedAlphabetically()

Debe mostrar las asignaturas de las que hay algún alumno matriculado, ordenadas alfabéticamente. La salida sería:

```
Asignaturas:
LM
PROGR
```

8.- showStudentsGrantsAndSum()

Debe mostrar la beca de cada alumno y además la suma de todas las becas. La salida sería:

```
Becas:
Germán Ginés: 2000
Baldomero: 0
Ana Guerra: 4000
Suma de becas: 6000
```

9.- getStudentsOlderThan20()

Debe retornar una lista con los nombres de los alumnos mayores de 20 años. El retorno debería la lista:

```
[Germán Ginés, Baldomero]
```

10.- showYoungestStudentName()

Debe mostrar el nombre de alumna más joven. La salida sería:

```
Alumno más joven: Ana Guerra
```

11.- showOldestStudentOlderThan23()

Debe mostrar el nombre del alumnos más veterano de entre los que tengan más de 23 años. La salida sería:

```
Alumno más veterano mayor de 23: No encontrado
```

12.- showStudentNamesWithCommasOrderedByAge()

Debe mostrar una cadena con los nombres de los alumnos separados por coma, ordenados por su edad. La salida sería:

```
Alumnos: Ana Guerra, Baldomero, Germán Ginés
```

13.- showStudentCountInEachGroup()

Debe mostrar el número de alumnos de cada grupo-clase, ordenados por nombre del grupo. La salida sería:

```
Número de alumnos en cada grupo:
1º CFGS DAM: 2 alumnos
1º CFGM SMR: 1 alumno
```

Intenta hacer que el listado salga ordenado por el nombre del grupo-clase, a ver si lo consigues.

14.- showGrantSummary()

Debe mostrar la estadística de becas de los alumnos, es decir, la beca máxima, la mínima y la media (haciendo todos los cálculos de una sola vez). La salida sería:

```
Estadística de becas:
Máxima: 4000, Mínima: 0, Media: 2000,00
```

15.- showAreAnyStudentUnderLegalAge()

Debe mostrar si hay algún alumno menor de edad. La salida sería:

```
¿Algún alumno menor de edad? Sí
```

16.- showAllStudentHaveGrant()

Debe mostrar si todos los alumnos tienen beca. La salida sería:

```
¿Todos los alumnos tienen beca?: No
```

17.- showFirstStudentWithoutGrant()

Debe mostrar el nombre del primer alumno que no tenga beca (grant == 0). La salida sería:

```
Nombre del primer alumno sin beca: Baldomero
```

18.- showHowManyStudentWithOrWithoutGrant()

Debe mostrar cuántos alumnos hay con beca y cuántos sin beca. La salida sería:

```
Alumnos con o sin beca
Sin beca: 1
Con beca: 2
```

19.- showNumberOfSubjectsOfEachStudent()

Debe mostrar el número de asignaturas de las que está matriculado cada alumnos. La salida sería:

```
Número de asignaturas de cada alumno:
Baldomero: 2
Ana Guerra: 1
Germán Ginés: 2
```

20.- showNumberOfPassersStudentsOfEachSubject() (nivel Pro)

Debe mostrar el número de alumnos aprobados en cada asignatura (mark >= 5). La salida sería:

```
Número de aprobados por asignatura:
PROGR - 3 aprobados
LM - 0 aprobados
```

Intenta hacer que el listado salga ordenado por nombre de la asignatura.