Ejercicios de repaso:

1) Un día, Caperucita Roja decide ir desde su casa hasta la de su abuelita, recolectando frutos del bosque del camino y tratando de hacer el paseo de la manera más segura posible. La casa de Caperucita está en un claro del extremo oeste del bosque, la casa de su abuelita en un claro del extremo este, y dentro del bosque entre ambas hay algunos otros claros. El bosque está representado por un grafo, donde los vértices representan los claros (identificados por un String) y las aristas los senderos que los unen. Cada arista informa la cantidad de árboles frutales que hay en el sendero. Caperucita sabe que el lobo es muy goloso y le gustan mucho las frutas, entonces para no ser capturada por el lobo, desea encontrar un camino que no pase por los senderos con cantidad de frutales >= 5. Como miembros de la sociedad Ayuda a Caperucita a Moverse (ACM), están llamados a ayudarla a encontrar ese camino, y cuando lo hayan hecho, colorín, colorado, este problema habrá terminado. Su tarea es definir una clase llamada BuscadorDeCamino, con una variable de instancia llamada “bosque” de tipo Grafo, que representa el bosque descripto e implementar un método de instancia con la siguiente firma:

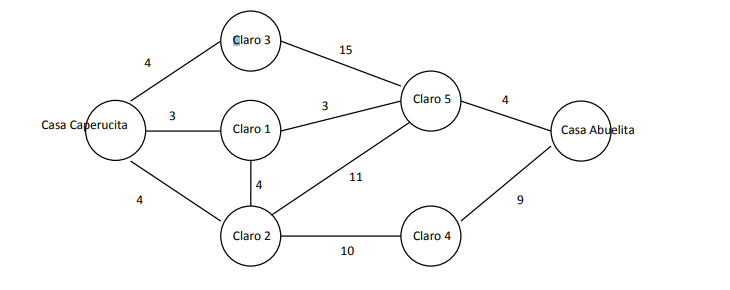
**Public ListaGenerica** recorridoMasSeguro () que devuelva un listado con los claros que conforman el recorrido encontrado. Si existe más de uno, devuelva uno de ellos. Si no existe devuelva un listado vacío.

Nota: La casa de caperucita debe ser buscada antes de comenzar a buscar el recorrido, y tiene la identificación: "Casa Caperucita".

**Caminos posibles:**

1) Casa Caperucita - Claro 1 - Claro 5 - Casa Abuelita.

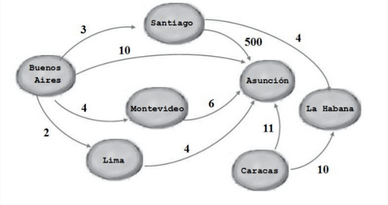
2) Casa Caperucita - Claro 2 – Claro 1 - Claro 5 - Casa Abuelita.



Si es un solo camino no es necesario desmarcar. Si llego a un claro donde no me puede conducir al destino, lo debo marcar para luego no volver a visitarlo. Si lo desmarco, puedo volver a visitarlo y nuevamente no me conduciría a ningún lado. Cada vértice se marca independientemente.

Es conveniente desmarcar cuando me pidan varios caminos o el mejor de todos los caminos posibles. En este caso el vértice marca y agrega a sus adyacentes.

2) Dado un Grado orientado y valorado positivamente, implemente un método que retorne una lista con el camino desde un origen a un destino. En caso de que haya más de un camino, devolver el que pase por menos ciudades.



3) Un día, Caperucita Roja decide ir desde su casa hasta la de su abuelita, recolectando frutos del bosque del camino y tratando de hacer el paseo de la manera más segura posible. La casa de Caperucita está en un claro del extremo oeste del bosque, la casa de su abuelita en un claro del extremo este, y dentro del bosque entre ambas hay algunos otros claros. El bosque está representado por un grafo, donde los vértices representan los claros (identificados por un String) y las aristas los senderos que los unen. Cada arista informa la cantidad de árboles frutales que hay en el sendero. Caperucita sabe que el lobo es muy goloso y le gustan mucho las frutas, entonces para no ser capturada por el lobo, desea encontrar un camino que no pase por los senderos con cantidad de frutales >= 5. Como miembros de la sociedad Ayuda a Caperucita a Moverse (ACM), están llamados a ayudarla a encontrar ese camino, y cuando lo hayan hecho, colorín, colorado, este problema habrá terminado! Su tarea es definir una clase llamada BuscadorDeCamino, con una variable de instancia llamada “bosque” de tipo Grafo, que representa el bosque descripto e implementar un método de instancia con la siguiente firma:

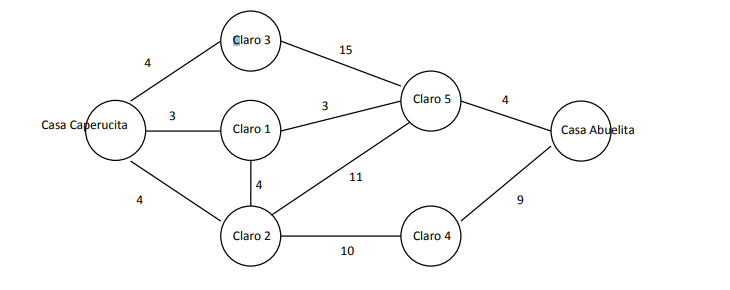
Public ListaGenerica recorridosMasSeguros () que devuelva un listado con los claros que conforman el recorrido encontrado. Si existe más de uno, devuelva uno de ellos. Si no existe devuelva un listado vacío. Nota: La casa de caperucita debe ser buscada antes de comenzar a buscar el recorrido, y tiene la identificación: "Casa Caperucita".

Cuando encuentre un camino, sigo buscando otros caminos posibles.

Caminos posibles:

1) Casa Caperucita - Claro 1 - Claro 5 - Casa Abuelita.

2) Casa Caperucita - Claro 2 – Claro 1 - Claro 5 - Casa Abuelita.



Cuando encuentre un camino válido lo almaceno en una lista.

4) buscar el camino más corto en cuanto a las longitudes. Recorro todos los vértices porque necesito comparar todas las combinaciones. El vértice marca a sus adyacentes, no se marcan independientemente.

