Prueba 1

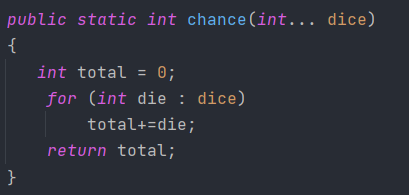
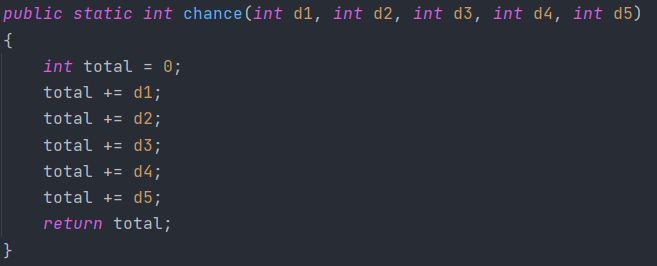
Nombre: Diego Garrido

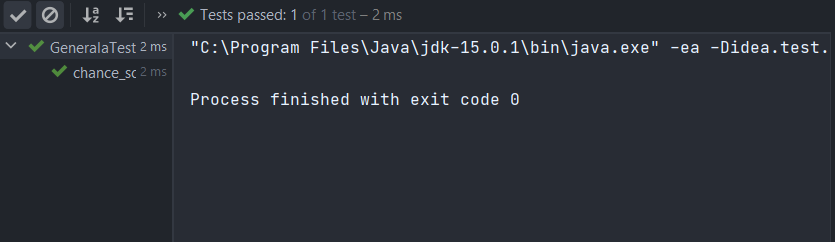
Profesor: Marcelo Espergel

Asignatura: Programación Avanzada

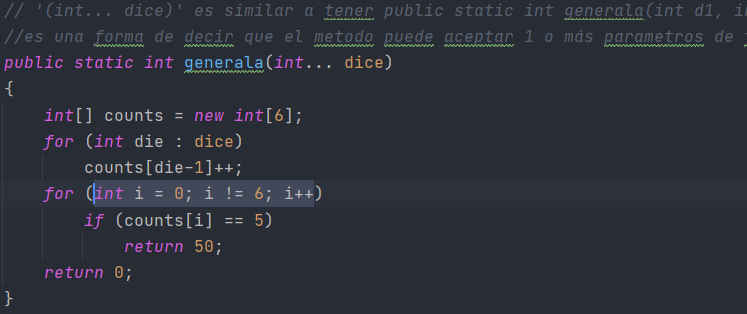
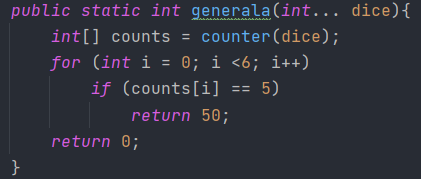
***Desarrollo:***

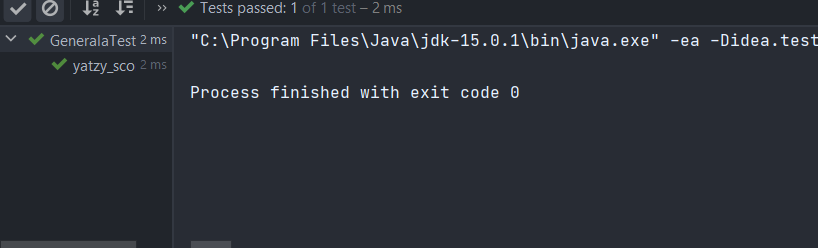
Refactoring método “chance”: este método obtiene la suma de todos los dados, pero lo asignaba dado por dado asique lo que hice fue con un for-each tome los dados que le daban al método y los sume a una variable.

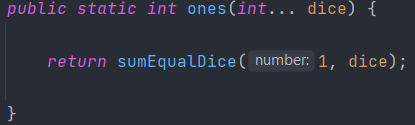
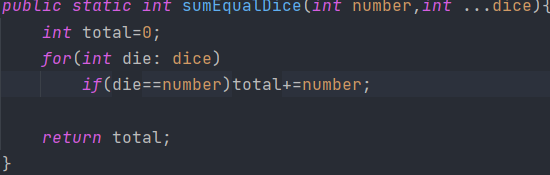


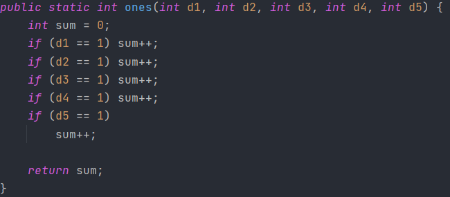


Refactoring método “generala”: este método en caso que todos los dados tengan el mismo valor obtiene 50 puntos, aquí implemente un método llamado “counter” el cual es un método contador, también cambie un símbolo para la eficiencia y elimine los comentarios.

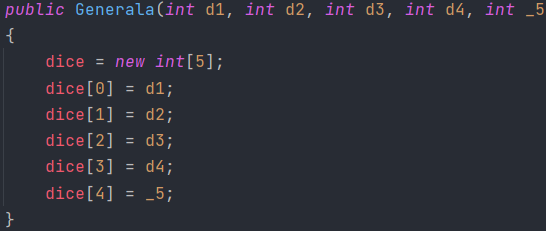
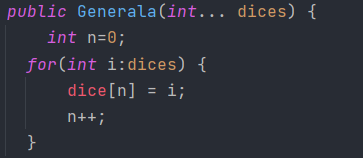


Refactoring métodos “ones, twos, threes”: aquí implemente un método llamado “sumEqualDice” el cual tomo como parámetro un numero el cual se esta evaluando y los números asignados a los dados, se compara el numero y si es asi se suma a una variable total para posteriormente ser retornada



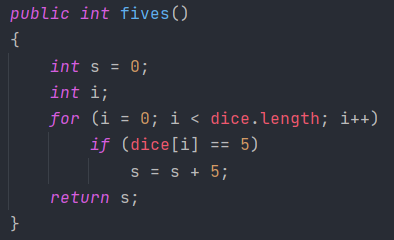
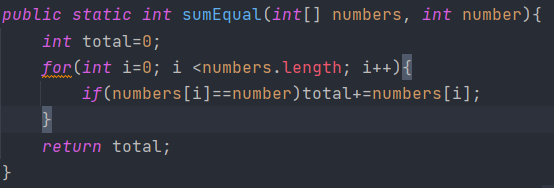
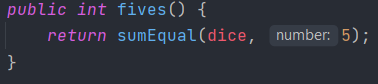
  

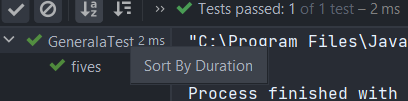
Refactoring método “Generala”: aquí asignaba valores de los dados a un arreglo uno por uno, asique las asigne con un for-each

No incluire test ya que este metodo no lo tiene, pero si se usa para los siguinetes

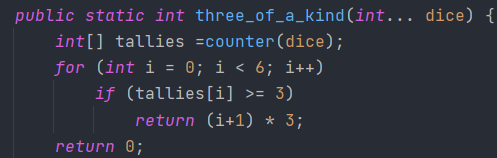
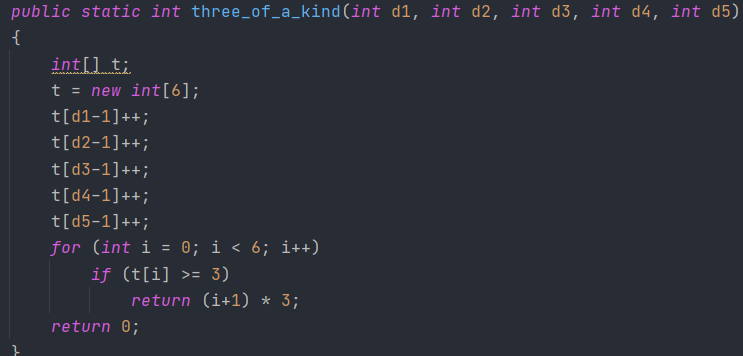
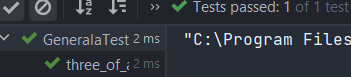
Refactoring métodos “fours, Fives, Sixes”: Para estos métodos cree un método llamado “sumEquals” el cual recibe el arreglo y un numero, este método con un for compara si la casilla contiene el mismo numero que el numero solicitado, si es así, se suma a una variable llamada total y la retorna.





Refactoring métodos “score pair , twopair, threes of kinds,four of kinds”:

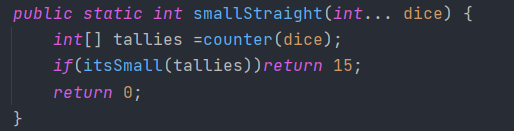
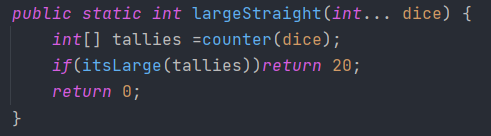
para estos métodos me di cuenta que hacían lo mismo en un principio, hacían un contador, así que implemente nuevamente el método counter, además de cambiar algunos símbolos en un for para la eficiencia.

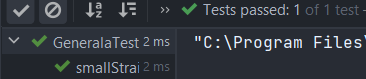
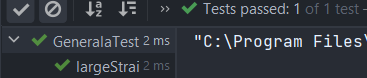
  

Refactoring métodos “smallStraight, largeStraight”:

Aquí implemente nuevamente el método “counter” y cree un nuevo método “itsSmall”, este método contiene la sentencia que determina si es small o no, si lo es retorna 15 si no 0.

Mismo procedimiento con “large” pero esta vez cree el método “itsLarge”.





Refactoring método “FullHause”: para este método implemente el método “counter” una vez más, optimice la inicialización de variables y sentencias, además, cambie el nombre de las variables para que fuesen mas descriptivos y para el posterior entendimiento.

