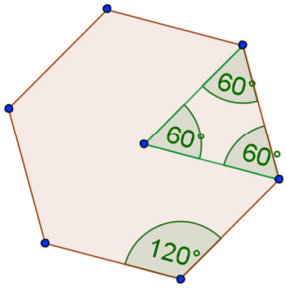
EF Informatik **Arbeitsblatt PolyForms 4**

Thema: Objektorientiertes Entwerfen / Modellieren / Implementieren mit Java

Arbeitsauftrag 4:



Die Klasse *PolyForms* soll jetzt um

- den privaten Dienst ***zeichneSechseck()*** sowie
- den öffentlichen Dienst ***zeichneSechseck(px: int, py:int, ps:int)***
  (px: x-Position, py: y-Position, ps: Größe/Seitenlänge)

erweitert werden. Ergänze das *UML*-Diagramm der Klasse entsprechend.

Teil a):

Entwerfe in Pseudocode zunächst die private Methode ***zeichneSechseck()***, die ein regelmäßiges Sechseck auf der Basis der Klassenattribute / Eigenschaften der Klasse *PolyForms* zeichnet (ähnlich dem zuvor erstellten Dienst *zeichneDreieck*).

Der Bezugspunkt für die Position des Sechsecks soll der Mittelpunkt des Sechsecks sein (Schnittpunkt der Seiten- und Winkelhalbierenden sowie Mittelpunkt von In- und Umkreis des Sechsecks, siehe Skizze). Das Attribut ***size*** gibt die Seitenlänge vor, das Attribut ***angle*** den Winkel gegenüber der Horizontalen.

Die Kennnummer ***shape*** für Sechsecke ist 2.

Teil b):

Übersetze dann den Entwurf in syntaktisch korrekten und vollständigen *Java-Quellcode* (auf Papier!). Dieser muß direkt mit dem in a) erstellten Pseudocode korrelieren!

Erstelle auf dieser Basis weiterhin den Quellcode für die öffentliche Methode *zeichneSechseck* mit der oben angegebenen Signatur direkt in *Java*. Sie soll bei Aufruf die direkte Vorgabe der Position und der Größe eines Sechsecks über ihre Aufrufparameter ermöglichen.

Teil c):

Nach der Diskussion / Vorstellung im Plenum erweitere in ***BlueJ*** die Klasse ***PolyForms*** um eine Implementation beider Methoden**.** Ergänze dann die Methode ***zeichne()*** um einen geeigneten Aufruf der Methode ***zeichneSechseck()*** in Abhängigkeit vom Wert des Attributes ***shape***.

Übersetze die Klasse und beseitige alle möglicherweise noch vorhandenen Syntaxfehler.

Erzeuge in *BlueJ* durch Aufruf des gewünschten Konstruktors eine Instanz der Klasse *PolyForms* und erprobe mit Hilfe des Kontextmenüs der Instanz (siehe auch Arbeitsblatt 2) auch die logische Korrektheit beider Methoden (Semantik) durch das Erzeugen verschiedener Ausprägungen von Sechsecken (Größe, Position, Winkel) mit Hilfe der *Set*-Methoden sowie der *zeichne()* und *loesche()*-Methode (entsprechend dem Vorgehen bei der *zeichneDreieck*-Methode).

Korrigiere, falls erforderlich, die Logik deiner Implementierung anhand der Musterlösung, sodass Sechsecke wie erwartet gezeichnet werden.

Stelle den Java-Quellcode der Methode ***zeichne()*** in die Cloud.