Даалгавар 1

Энэхүү даалгаврын зорилго нь өгөгдсөн илэрхийллүүдийг өөр өөр дүрмийн дагуу үйлдлийн эрэмбэ, хамаарлыг үнэлнэ. Хүснэгт 1-д хоёр багц дүрмийг (Rules) тодорхойлсон. "Prec" багана нь эрэмбэ (precedence), "Assoc" баганын "L" нь зүүн-хамаарал (left-associative), "R" нь баруун-хамаарал (left-associative)

Operator	Rules 1		Rules 2		Type	
	Assoc.	Prec.	Assoc.	Prec.	Operands	Results
**	R	2	R	2	numeric	numeric
*	L	3	R	3	numeric	numeric
/	L	3	R	3	numeric	numeric
+	L	4	R	3	numeric	numeric
-	L	4	R	3	numeric	numeric
>>	L	5	R	4	numeric	numeric
<<	L	5	R	4	numeric	numeric
>	L	6	L	5	numeric	boolean
<	L	6	L	5	numeric	boolean
!=	L	7	L	6	numeric	boolean
&&	L	8	L	7	boolean	boolean
||	L	8	L	7	boolean	boolean

Rules 1-ээр үнэлэх илэрхийллүүд

1. `5 ** 2 >> 6 / 3`
2. `2 ** 2 ** 3 / 4 + 3 * 2`
3. `12 - 2 * 4 != 1 && 9 + 10 / 2 > 12`

Rules 2-ээр үнэлэх илэрхийллүүд

4. `3 * 2 + 4 << 3 << 1`
5. `2 * 4 + 9 < 20 || 45 / 5 + 4 != 5`
6. `5 * 3 ** 2 + 5 > 6 ** 3 >> 4 / 2`

Санамж: Тусдаа операторуудын утга нь Java хэл дээрхтэй ижил байна. Нэмж хэлэхэд, '**' нь экспоненциал (зэрэгт дэвшүүлэх) үйлдлийг илэрхийлнэ (жишээ нь, 2 ** 3 нь 8). Илэрхийллийг тооцоолж үр дүнг харуулахын тулд Java синтаксийг ашиглана (жишээ нь, тавын тоо 5, логикийн утга нь true, false).

Даалгавар 2

Энэ даалгавар нь Python дээр модны нэвтрэлтийг хэрэгжүүлнэ. Модыг хэрэгжүүлсэн Python класс "/task 1" хавтсанд өгсөн бөгөөд зангилаа бүр тэмдэгт мөрийн жагсаалтыг

хадгалдаг. Таны даалгавар бол модны бүх зангилааны бүх мөрийг давтахын тулд Tree классын

$$iterate\_strings\_in\_all\_nodes()$$

методыг хөгжүүлэх юм. Давталтын дарааллыг (infix, prefix, postfix) нь өөрөө сонгоно.

Даалгавар 3

Энэ даалгаврын хувьд Scala-д дараах гурван тайл-рекурсив (tail-recursive) функцийг хэрэгжүүлнэ:

1. filterGreaterThanBase, массивын элементүүд суурь бүхэл тооноос их эсэхийг шалгадаг.
2. isDaffodil, бүхэл тоо нь нарциссист (эсвэл Armstrong) тоо эсэхийг шалгадаг.
3. countLetters, Үсэг бүр мөрөнд хэдэн удаа тохиолдож байгааг тоолдог.

Санамж: Даалгаврыг Scala-д хэрэгжүүлнэ (хувилбар = 3.2.2). Та Scala-г эндээс авна. Функцүүдийг тайл-рекурсив байдлаар хэрэгжүүлэх ёстой. Товчхондоо, өөрийгөө сүүлчийн зааврааараа дууддаг бол функц нь тайл-рекурсив байна. Scala нь компиляторт тухайн функцийг тайл-рекурсив гэдгийг тодорхойлох *@tailrec* тусгай тэмдэглэгээ хэрэглэдэг. Энэ функц нь алдааг эрт илрүүлэхэд тусална, учир нь tail-recursive гэж тэмдэглэсэн боловч компилятор таних боломжгүй бол хөрвүүлэлтийн үеийн алдаа өгнө.

Эхлэл

Энэ даалгаварт ашиглах загвар Scala төслийн загварыг /task 2 хавтаснаас олох болно. 'src' хавтас нь энэ даалгаварт ашиглах ёстой нэг ширхэг эх файлыг агуулдаг. Эх кодын файл (Main.scala) ашиглан, зөвхөн уг загварын дагуу хэрэгжүүлээрэй. Main.scala-д гурван функцийн их биеийг бичиж хэрэгжүүлнэ. Зөвхөн Scala стандарт сангуудыг ашиглах боломжтой.

Тестчилэл

Гурван функцийн зөв оролт-гаралтын хосуудын жишээ:

Input	Output
`filterGreaterThanBase(Array(5, 21, 17, 4, 10, 15), 12, Array.empty[Int])`	`21, 17, 15`
`isDaffodil(200, 0)`	`false`
`isDaffodil(153, 0)`	`true`
`countLetters("aadabcbbbcc", Map.empty[Char, Int])`	`Map(a -> 3, d -> 1, b -> 4, c -> 3)`

ТАНД АМЖИЛТ ХҮСЬЕ!