Kaba - Programming Language Documentation

Kaba is a parody programming language intended to imitate a furious Turkish uncle yelling and swearing at a malfunctioning TV.

1 General Rules

1. Every keyword must be as insulting as possible.
2. Any characters that are not part of the Turkish alphabet, a space, a newline, or an apostrophe (') are treated as comments. This includes digits, which are disallowed in code. See the "Numbers" section for how to represent numeric values.
3. The syntax closely follows Turkish grammar rules. Consequently, there is a special class of keywords called **suffixes**, explained in detail later.
4. Kaba is **case-sensitive**.

2 Keywords

In this first version of Kaba, there are a total of 37 keywords:

```
Bok
Torbası
boku
kat
sök
çak
yar
aynı
tersi
koymuş
koymamış
ve
ya
siktiyse
sikmediyse
sikerken
```

```
sallayarak
çök
kus
denesene
sıçarsa
rezalet
patlat
sok
şöyle
böyle
demiş
dememiş
hadi
siktir
virgül
amk
amq
aq
amık
amınakoyum
suffixes
```

3 Primitive Data Types

There are four primitive types in Kaba:

- **Düz**: 64-bit signed integer
- **Sulu**: 64-bit double-precision floating-point
- **Net**: Boolean
- **Saçma**: String

You can also create a dynamic array of any type by adding the keyword **Torbası** after the type specifier.

Example:

```
Düz Bok Torbası
```

To access an element of a **Bok Torbası**, use the following syntax:

```
(Variable Name)(Genitive Suffix) (Index)(Ordinal Suffix) boku
```

Example:

```
boktorbası'nın bir'inci boku
```

4 Suffixes

In Kaba, **suffixes** are characters from the Turkish alphabet that follow an apostrophe (’). They are divided into **eight** groups:

- Accusative
- Dative
- Ablative
- Genitive
- Possessive
- Comitative
- Locative
- Ordinal

Suffixes must adhere to Turkish **vowel harmony** rules.

5 Statements

All statements in Kaba must end with one of the following keywords:

```
amk
amq
aq
amık
amınakoyum
```

5.1 Assignment Statements

To assign a value to a variable, use the **sok** keyword with the following syntax:

```
(Variable Name)(Dative Suffix) (Expression)(Accusative Suffix) sok amk
```

Example:

```
bokbir’e binbir’i sok amk
```

6 Numbers

Kaba **forbids** the use of numerical digits. All numbers must be written out **in Turkish words** with correct grammar. There are two categories of numeric expressions:

- **Düz Bok (Integers)**
- **Sulu Bok (Floating-Point Numbers)**

6.1 Düz Bok (Integers)

Write integers as a continuous string of Turkish number words, without spaces or hyphens.

Examples:

```
1: bir
10: on
1536: binbeşyüzotuzaltı
-101: eksiyüzbir
```

6.2 Sulu Bok (Floating-Point Numbers)

Use the word **virgül** to indicate the decimal point. The digits after **virgül** must be spelled out as a single whole number, not digit-by-digit.

6.2.1 Important Rules:

1. The digits after the decimal point must **not end with zero**, unless the only digit after the decimal is zero (e.g., `birvirgülsıfır` is valid, `ikivirgüldoksan` is not).
2. If there are **leading zeros** immediately after the decimal point (e.g. `0.0001235`), each zero must be individually written out as `sıfır` (without separators).

Examples:

```
1.0: birvirgülsıfır
2.3: ikivirgülüç
2.30: Not allowed
3.01: uçvirgülsıfırbir
1009.0001235: bindokuzvirgülsıfırsıfırsıfırbinikiyüzotuzbeş
```

7 Operators

There are **10** operators in Kaba. Operands must be used with **proper suffixes**. Below are the operators and their general formats.

7.1 kat (addition)

Format (Düz):

```
(Düz Bok)(Accusative) (Düz Bok)(Dative) kat
```

Format (Sulu):

```
(Sulu Bok)(Accusative) (Sulu Bok)(Dative) kat
```

Example:

```
bokbir'i bokiki'ye kat
```

7.2 sök (subtraction)

Format (Düz):

`(Düz Bok)(Ablative) (Düz Bok)(Accusative) sök`

Format (Sulu):

`(Sulu Bok)(Ablative) (Sulu Bok)(Accusative) sök`

Example:

`bokbir'den bokiki'yi sök`

7.3 çak (multiplication)

Format (Düz):

`(Düz Bok)(Accusative) (Düz Bok)(Dative) çak`

Format (Sulu):

`(Sulu Bok)(Accusative) (Sulu Bok)(Dative) çak`

Example:

`bokbir'i bokiki'ye çak`

7.4 yar (division)

Format (Düz):

`(Düz Bok)(Accusative) (Düz Bok)(Dative) yar`

Format (Sulu):

`(Sulu Bok)(Accusative) (Sulu Bok)(Dative) yar`

Example:

`bokbir'i bokiki'ye yar`

7.5 aynı (equality)

Format (Düz):

`(Düz Bok)(Comitative) (Düz Bok) aynı`

Format (Sulu):

`(Sulu Bok)(Comitative) (Sulu Bok) aynı`

Format (Net):

`(Net Bok)(Comitative) (Net Bok) aynı`

Format (Saçma):

`(Saçma Bok)(Comitative) (Saçma Bok) aynı`

Example:

`bokbir'le bokiki aynı`

7.6 koymuş (greater than)

Format (Düz):

`(Düz Bok) (Düz Bok)(Dative) koymuş`

Format (Sulu):

`(Sulu Bok) (Sulu Bok)(Dative) koymuş`

Example:

`bokbir bokiki'ye koymuş`

7.7 koymamış (less than)

Format (Düz):

`(Düz Bok) (Düz Bok)(Dative) koymamış`

Format (Sulu):

`(Sulu Bok) (Sulu Bok)(Dative) koymamış`

Example:

`bokbir bokiki'ye koymamış`

7.8 tersi (not)

Format (Net):

`(Net Bok)(Genitive) tersi`

Example:

`bokbir'in tersi`

7.9 ve (and)

Format (Net):

`(Net Bok) ve (Net Bok)`

Example:

`bokbir ve bokiki`

7.10 ya (or)

Format (Net):

```
(Net Bok) ya (Net Bok)
```

Example:

```
bokbir ya bokiki
```

8 Parentheses, Brackets, and Quotes

8.1 şöyle ... böyle: ()

Format:

```
şöyle (Expression) virgül (Expression) virgül ... böyle
```

Example:

```
şöyle bokbir virgül bokiki böyle
```

8.2 hadi ... siktir: {}

Format:

```
hadi
  (Statement)
  (Statement)
  ...
siktir
```

Example:

```
hadi
  bokbir'e bokiki'yi sök amk
siktir
```

8.3 demiş ... dememiş: ""

Format:

```
demiş (String) dememiş
```

Example:

```
demiş This is a test String dememiş
```

9 Declarations

All variables must be declared before use with the **çök** keyword.

Format:

```
(Class Bok)(Ablative Suffix) (Variable Name) çök amk
```

Example:

```
Düz Bok'tan bokbir çök amk
```

10 Conditional Statements

Kaba provides **siktiyse** (if) and **sikmediyse** (else).

Format:

```
(Expression) siktiyse
hadi
  (Statement)
siktir
sikmediyse hadi
  (Statement)
siktir
```

Example:

```
bokbir bokiki'yle aynı siktiyse hadi
  bokbir'e sıfır'ı sok amk
siktir sikmediyse
hadi
  bokiki'ye sıfır'ı sok amk
siktir
```

11 Loops

11.1 sikerken: while

Format:

```
(Expression) sikerken hadi
  (Statement)
siktir
```

Example:

```
bokbir sikerken hadi
  bokiki'ye bir'i kat amk
siktir
```

11.2 sallayarak: for

Format:

```
(Variable Name)(Accusative Suffix) (Bok Torbası)(Locative Suffix) sallayarak hadi
  (Statement)
siktir
```

Example:

```
sallayanbok'u boktorbası'nda sallayarak hadi
  bokbir'e şöyle bokbir'i sallayanbok'a kat böyle'yi sok amk
siktir
```

12 Functions

Functions are defined by specifying their return type and the types of their arguments. There is no **void** return type in Kaba; every function must have exactly one **kus** (return) statement. If there are multiple arguments, they are separated by the **virgül** keyword.

12.1 Definition

Format:

```
(Return Type) (Function Name) şöyle (Arguments) böyle hadi
  (Statement)
  (Expression)(Accusative Suffix) kus amk
siktir
```

Example:

```
Düz Bok'tan kuvvet şöyle Düz Bok bokbir virgül Düz Bok bokiki böyle hadi
  Düz Bok'tan boküç çök amk
  Düz Bok'tan bokdört çök amq
  boküç'e bir'i sok aq
  bokdört'e bokiki'yi sok amınakoyum

  şöyle bokdört sıfır'a koymuş böyle sikerken
  hadi
    boküç'e şöyle boküç'ü bokbir'e çak böyle'yi sok amk
    bokdört'e şöyle bokdört'ten bir'i sök böyle'yi sok aq
  siktir

  böküç'ü kus amk
siktir
```

12.2 Call

Format:

```
(Function Name) şöyle (Arguments) böyle
```

Example:

```
kuvvet şöyle iki virgül üç böyle
```

13 Classes

Kaba supports composite types through **classes**. There are **no constructor functions**; objects must be initialized externally after declaration. Class definitions only contain **attribute declarations** and **method definitions**.

13.1 Definition

Format:

```
(Class Name) Bok hadi
  (Attribute Declaration)
  (Method Definition)
siktir
```

Example:

```
Öğrenci Bok hadi
  Saçma Bok'tan isim çök amk
  Saçma Bok'tan isimNe söyle böyle hadi
    isim'i kus aq
  siktir
siktir
```

13.2 Instantiation

Objects are created using the variable declaration syntax.

Format:

```
(Class Name Bok)(Ablative Suffix) (Variable Name) çök amk
```

Example:

```
Öğrenci Bok'tan öğrenci çök amk
```

13.3 Attribute and Method Access

Use **Genitive** and **Possessive** suffixes to access attributes or methods.

Format:

```
(Object Name)(Genitive Suffix) (Attribute/Method)(Possessive Suffix)
```

Examples:

```
öğrenci'nin isim'i
öğrenci'nin isimNe'si söyle böyle
```

14 Exceptions

Kaba's error handling is minimal. There is only one Exception type: **Saçma Bok**.

There are four keywords for exceptions:

- **denesene** (try)
- **sıçarsa** (except)
- **patlat** (throw)
- **rezalet** (the exception object, always Saçma Bok)

14.1 Format

```
denesene hadi
  (Statement)
siktir
sıçarsa hadi
  (Statement)
siktir
(Exception)(Accusative Suffix) patlat amk
```

Example:

```
denesene hadi
  Düz Bok'tan bokbir çök amk
siktir
sıçarsa hadi
  rezalet'i patlat amk
siktir
```