Leírás

```
double evalfunc(String input, long T)
```

Egy egyszerű top-down parsert valósít meg, mely az alábbi formájú **kifejezés**eket tudja kiértékelni.

kifejezés= T| R | **szám** |
(**kifejezés**)
kifejezés infixop kiejezés |
s (**kiejezés**) |
p (**kiejezés** , **kiejezés** , **kifejezes**) |
m (**kiejezés** , **kiejezés** , **kifejezes**) |
c (**kiejezés** , **kiejezés** , **kifejezes** , **kiejezés** , **kiejezés** , **kifejezes**) |

infixop = ^| * | / | + | -

preifxop= s | p | m | c

szám = Minden olyan Java `String`, amit az `isNumber` függvény elfogad.

```
public static boolean isNumber(String n) {
        try
        {
              Double.parseDouble(n);
        }
        catch(NumberFormatException nfe)
        {
              return false;
        }
        return true;
}
```

Tehát 64 bites lebegőpontos számokat használhatunk, a törteket tizedesponttal tagolhatjuk, vagy normálalakban is felírhatjuk.

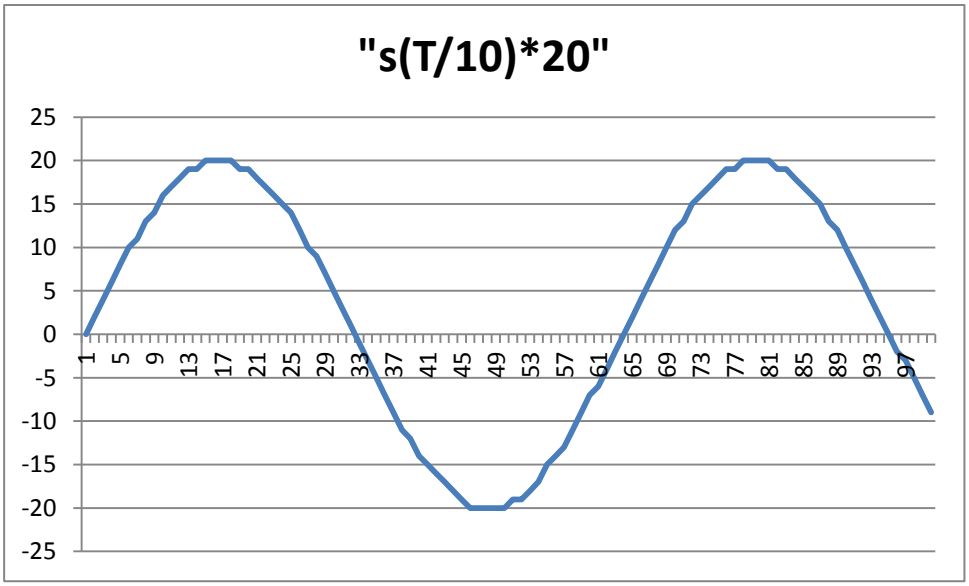
Az aritmetikai operátorok jelentése a szokásos. A T `long` típusú változó a függvény bemenő paramétere, R a [0;1] intervallumból egyenletes eloszlással generálódó véletlenszám.

A prefix operátorok jelentése a következő:

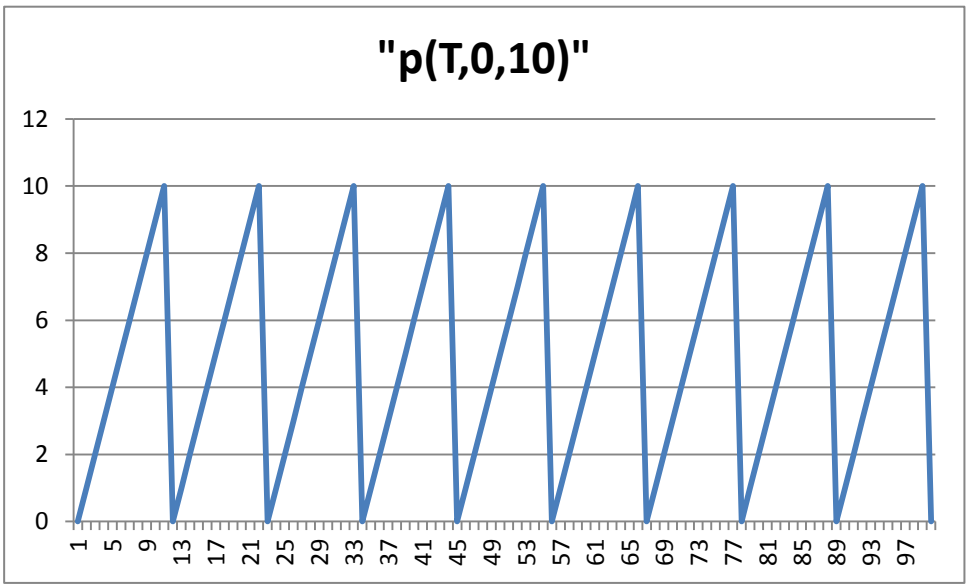
sin(kif), a szinusz függvény szokásos definíciójának megfelelően.

p(kif,start,end) Periodikus függvény a kif **kifejezés** [start;end] intervallumbeli értékeit ismételgeti

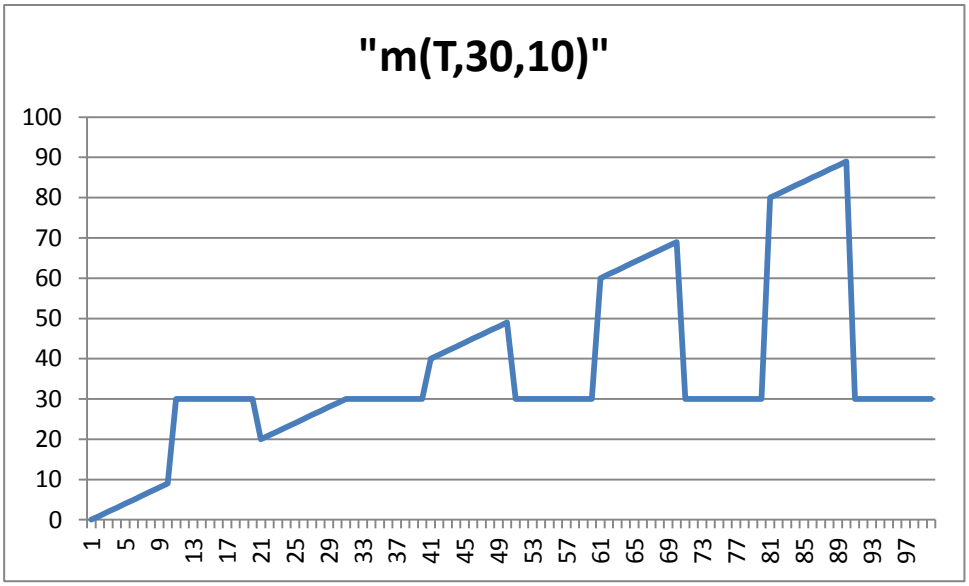
s(kif) Értéke azonos sin(kif)-fel, a szinusz függvény szokásos definíciójának megfelelően.
Példa:z



p(kif,start,end) Periodikus függvény, a kif **kifejezés** [start;end] intervallumbeli értékeit.
Példa:



m(kif1,kif2,width) Két függvény merge-elése. Periodikusan váltakozva (width periódussal) kif1 és kif2 értékeit veszi fel.
Példa:



c(kif1,kif2,s1,e1,s2,e2) A két függvény, kif1 [s1;e1] és kif2 [s2;e2] intervallumának konkatenálása.
Példa: