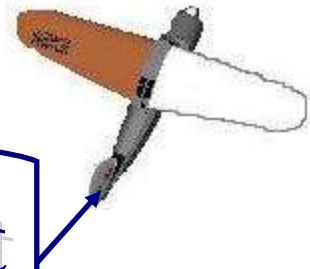
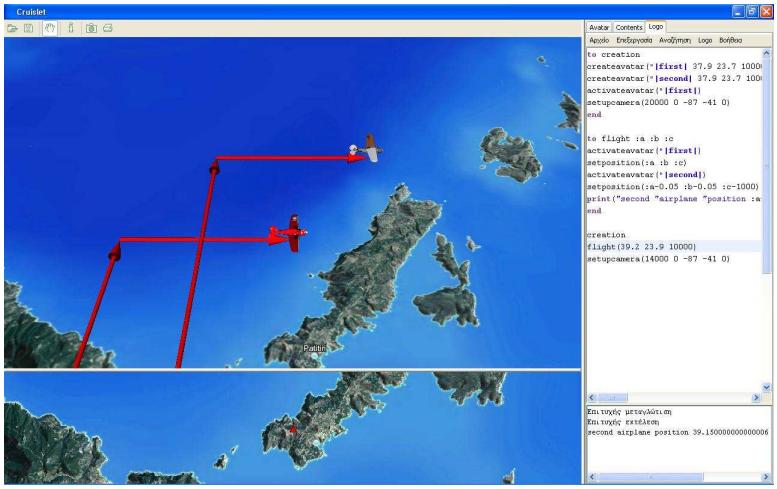
Εργαστήριο Εκπαιδευτικής Τεχνολογίας

Πανεπιστήμιο Αθηνών



Η Logo του Cruislet



Επιμέλεια: *Έφη Αλεξοπούλου*

Εισαγωγή

Η Logo του Cruislet είναι βασισμένη στη γλώσσα προγραμματισμού Turtletracks που δημιουργήθηκε από τον Daniel Azuma. Περισσότερες πληροφορίες για τη γλώσσα αυτή μπορείτε να βρείτε στην ιστοσελίδα: http://turtletracks.sourceforge.net/.

Η δομή του κειμένου που ακολουθεί, περιγράφει το λεξιλόγιο της Logo του Cruislet και περιλαμβάνει:

- Την γραμμή σύνταξης της εντολής ή της συνάρτησης.
- Επεξηγηματικό κείμενο για τη λειτουργία της εντολής ή της συνάρτησης.
- Παράδειγμα χρήσης, όπου με γκρι σκίαση είναι ο κώδικας, ο οποίος εφόσον μεταφερθεί στον «Συντάκτη Πηγαίου Κώδικα» του Cruislet και εκτελεστεί, έχει ως αποτέλεσμα, στο παράθυρο «Έξοδος Μηνυμάτων», το περιεχόμενο του πλαισίου που ακολουθεί τον κώδικα.

Σημείωση: Για την εκτέλεση εντολών στη Logo του Cruislet, χρησιμοποιείται το πλήκτρο Insert.

Σύμβολα στη Logo

Υπάρχουν δύο σημαντικά σύμβολα στην logo. Τα διπλά εισαγωγικά «"» και η ανω και κάτω τελεία «:».
Τα *διπλά εισαγωγικά* δηλώνουν πως ό,τι ακολουθεί πρέπει να ληφθεί ως έχει από την logo, επομένως, να μην γίνει κανένας υπολογισμός πάνω σε αυτό. Μια λέξη, για παράδειγμα, αρχίζει με διπλά εισαγωγικά και τελειώνει με το χαρακτήρα του κενού (space). Αν θέλουμε να ορίσουμε μια σειρά από χαρακτήρες που να περιέχει και κενά, θα πρέπει να χρησιμοποιήσουμε την κάθετη γραμμή, "|", για να την οριοθετήσουμε.

```
print("Cruislet)
print("|Hello world|)
```

- *Cruislet*
- *Hello world*

Η *άνω και κάτω τελεία* χρησιμοποιείται για προσπελάσουμε την τιμή μιας μεταβλητής (βλέπε τις εντολές make και localmake). Στο παράδειγμα οι διαδοχικές άνω και κάτω τελείες επιτρέπουν να προσπελαύνονται τιμές μεταβλητών, που τα ονόματά τους είναι περιεχόμενα άλλων μεταβλητών.

```
make "y 10
make "x "y
make "z "x
print(:z)
print(::z)
print(:::z)
```

- *x*
- *y*
- *10*

Η *εισαγωγή σχολίων*, στον Συντάκτη του Πηγαίου Κώδικα, γίνεται αρχίζοντας, την γραμμή των σχολίων, με τον χαρακτήρα «;».

Εντολές πλοήγησης

Στο περιβάλλον του Cruislet ο χρήστης μπορεί να πλοηγηθεί στον τρισδιάστατο γεωγραφικό χώρο. Το αντικείμενο (avatar) που μετατοπίζεται στο χώρο είναι αεροπλάνο ή ελικόπτερο.

Δημιουργία / Διαγραφή

Ο χρήστης πρέπει αρχικά να δημιουργήσει το αντικείμενο στη Logo.

createavatar(lat long height model_name)
Το model-name μπορεί να πάρει τις τιμές "|Plane 1|, "|Plane 2|, "|Helicopter|. Τα lat, long μπορούν να πάρουν τιμές μέσα στα πλαίσια του γεωγραφικού χώρου της Ελλάδας.

```
createavatar(37.4 26.2 10000 "|Plane 1|)
```

Στην περίπτωση που ο χρήστης θέλει να ονομάσει το αντικείμενο έτσι ώστε να μπορεί να αναφέρεται σε αυτό σε επόμενες εντολές, η εντολή createavatar συντάσσεται ως εξής.

createavatar (name lat long height model_name)

```
createavatar("|MyAirplane| 36 25 5000 "|Helicopter|)
```

Η εντολή createavatar μπορεί να γραφεί και ως creav, που είναι η συντόμευση της εντολής.

removeavatar()
Η εντολή αυτή διαγράφει το αντικείμενο που είναι ενεργοποιημένο. Στην περίπτωση που υπάρχουν παραπάνω από ένα αντικείμενα στη σκηνή, τότε για να διαγραφεί ένα συγκεκριμένο αντικείμενο, πρέπει να χρησιμοποιηθεί το όνομα του αντικειμένου αυτού.

```
removeavatar("|MyAirplane|)
```

Η εντολή removeavatar μπορεί να γραφεί και ως remav, που είναι η συντόμευση της εντολής.

Ενεργοποίηση

Για να αναφερθεί ο χρήστης σε ένα αντικείμενο (π.χ. αεροπλάνο) και να μπορεί να δώσει εντολές σε αυτό, πρέπει να το ενεργοποιήσει. Η ενεργοποίηση μπορεί να γίνει σε ένα αντικείμενο το οποίο έχει πριν δημιουργηθεί.

activateavatar(name)

```
activateavatar("|MyAirplane|)
```

Η εντολή activateavatar μπορεί να γραφεί και ως actav, που είναι η συντόμευση της εντολής.

Μετατόπιση

Η μετατόπιση του αντικειμένου στον τρισδιάστατο χώρο είναι μπορεί να γίνει με δύο τρόπους:

1. Μέσω των γεωγραφικών συντεταγμένων

setposition(lat long height)
Θέτει το αντικείμενο στη θέση που καθορίζεται από την είσοδο της εντολής, δηλαδή στις συντεταγμένες lat, long και height.

```
setposition(37 25 2500)
```

Η εντολή setposition μπορεί να γραφεί και ως setpos, που είναι η συντόμευση της εντολής.
Για να βρει ο χρήστης τη θέση του αντικειμένου που είναι ενεργοποιημένο εκείνη τη στιγμή, χρησιμοποιείται η εντολή:

outputposition()
Το αποτέλεσμα της εντολής εμφανίζεται στην «Έξοδο Μηνυμάτων». Η εντολή outputposition μπορεί να γραφεί και ως oppos, που είναι η συντόμευση της εντολής.

2. Μέσω των σφαιρικών συντεταγμένων

setdirection(theta fi r)
Το αντικείμενο στρέφεται κατά τις γωνίες theta και fi, και μετατοπίζεται κατά r (σε μέτρα).

```
setdirection(45 90 5500)
```

Η εντολή setdirection μπορεί να γραφεί και ως setdir, που είναι η συντόμευση της εντολής.
Για να βρει ο χρήστης τη μετατόπιση του αντικειμένου που είναι ενεργοποιημένο εκείνη τη στιγμή, χρησιμοποιείται η εντολή:

outputdirection()

Το αποτέλεσμα της εντολής εμφανίζεται στην «Έξοδο Μηνυμάτων». Η εντολή outputdirection μπορεί να γραφεί και ως opdir, που είναι η συντόμευση της εντολής.

stopavatar()
Η εντολή αυτή σταματά τη μετατόπιση του αντικειμένου που είναι ενεργοποιημένο. Στην περίπτωση που υπάρχουν παραπάνω από ένα αντικείμενα στη σκηνή, τότε για να διαγραφεί ένα συγκεκριμένο αντικείμενο, πρέπει να χρησιμοποιηθεί το όνομα του αντικειμένου αυτού.

```
stopavatar("|MyAirplane|)
```

Η εντολή stopavatar μπορεί να γραφεί και ως stopav, που είναι η συντόμευση της εντολής.

Κατά τη μετατόπιση, το αντικείμενο αφήνει ένα ίχνος πίσω του που έχει τη μορφή διανύσματος. Στην περίπτωση που ο χρήστης θέλει να διαγράψει το ίχνος που αφήνει το αντικείμενο, χρησιμοποιείται η παρακάτω εντολή.

cleartrail
Η εντολή claertrail μπορεί να γραφεί και ως cltr, που είναι η συντόμευση της εντολής.

Κάμερα

setcamera(distance azim pospitch orientpitch orientyaw)
Θέτει τιμές στις αντίστοιχες ιδιότητες της κάμερας.

```
setupcamera(15000 0 -87 -41 0)
```

Η εντολή setcamera μπορεί να γραφεί και ως cam, που είναι η συντόμευση της εντολής. Οι εντολές που ακολουθούν χρησιμοποιούνται όταν θέλουμε να αλλάξουμε συγκεκριμένες ιδιότητες της κάμερας. Κάθε εντολή ακολουθείται από τη συντόμευση που μπορεί να χρησιμοποιηθεί αντί αυτής.

camdist setcameradistance(distance)

setcameraazimuth(azim)
Συντόμευση: camazim

setcamerapospitch(pospitch)
Συντόμευση: camppitch

setcameraorientationpitch(orientpitch)
Συντόμευση: camopitch

setcameraorientationyaw(orientyaw)
Συντόμευση: camoyaw

Εντολές εκχώρησης τιμών

Τα ονόματα των μεταβλητών ή των σταθερών, είναι λέξεις (συμβολοσειρές) και αρχίζουν πάντα με διπλά εισαγωγικά.

constant <ConstantName> value
Ορίζεται σταθερά με όνομα ConstantName και αποδίδεται τιμή (value) σε αυτή. Οι σταθερές δεν προβλέπεται να μεταβληθούν κατά την διάρκεια εκτέλεσης του προγράμματος.

```
constant "c 300000000 ; παράδειγμα σταθερής,  η ταχύτητα του φωτός c.
print(c)
```

- 300000000

localmake <VariableName> value
Ορίζεται μια τοπική μεταβλητή. Έχει ισχύ μόνο μέσα στην διαδικασία που έχει οριστεί και σε όλες τις υποδιαδικασίες που καλούνται από αυτή. Εκτός της διαδικασίας, στην οποία ορίζεται η τοπική μεταβλητή, δεν υφίσταται μεταβλητή "a" και επομένως το αποτέλεσμα της εντολής print(:a) είναι null (το κενό όρισμα). Το περιεχόμενο της μεταβλητής προσπελαύνεται με την σύνταξη :a.

```
to try ;ορίζεται η διαδικασία
```

```
localmake "a 123
print(:a)
end
try ;εκτελείται η διαδικασία
print(:a)
```

- *123*
- *null*

make <VariableName> value

Ορίζεται μια καθολική (global) μεταβλητή. Η τιμή της διατηρείται σε όλες τις υποδιαδικασίες του προγράμματος.

```
make "myVariable 12
print(:myVariable)
```

- *12*

Δομές δεδομένων

Στην γλώσσα Logo οι δομές δεδομένων είναι λίστες (list), λέξεις (word) ή πίνακες (array).
Μια *λίστα* είναι μια διατεταγμένη συλλογή αντικειμένων (των μελών της λίστας), το μήκος της οποίας είναι δυνατόν να μεταβάλλεται δυναμικά.
Μια *λέξη* είναι μια συμβολοσειρά από γράμματα και αριθμούς. Αρχίζει πάντα με διπλά εισαγωγικά.
Ο *πίνακας* είναι μια δομή με σταθερό μήκος, το οποίο καθορίζεται ταυτόχρονα με την κατασκευή του.

Λίστες (Lists) και Λέξεις (Words)

count(list), count(word)

Επιστρέφει το μήκος της συμβολοσειράς (word), όταν το όρισμα είναι μια συμβολοσειρά ή το πλήθος των μελών της λίστας, όταν το όρισμα είναι μια λίστα.

```
print(count(["a "b "c "d]))
print(count("Cruislet))
```

- *4*
- *8*

length(list), length(word)

Επιστρέφει το μήκος της συμβολοσειράς, όταν το όρισμα είναι μια συμβολοσειρά ή το πλήθος των μελών της λίστας, όταν το όρισμα είναι μια λίστα.

```
print(length(["a "b "c "d]))
print(length("Cruislet))
```

- *4*
- *8*

butfirst(list), butfirst(word)

Επιστρέφει την λίστα, που εισάγεται ως όρισμα, εκτός από το πρώτο μέλος της.

```
print(butfirst("Cruislet))
print(butfirst([1 2 3 5 6]))
```

- *ruislet*
- *[2, 3, 5, 6]*

butlast(list), butlast(word)

Επιστρέφει την λίστα, που εισάγεται ως όρισμα, εκτός από το τελευταίο μέλος της.

```
print(butlast("Cruislet))
print(butlast([1 2 3 5 6]))
```

- *Cruisle*
- *[1, 2, 3, 5]*

first(list), first(word)

Επιστρέφει το πρώτο μέλος της λίστας ή της λέξης, που εισάγεται ως όρισμα.

```
print(first("Cruislet))
print(first([1 2 3 5 6]))
```

- *C*
- *1*

last(list), last(word)

Επιστρέφει το τελευταίο μέλος της λίστας, που εισάγεται ως όρισμα.

```
print(last("Cruislet))
print(last([1 2 3 5 6]))
```

- *t*
- *6*

item(number list), item(number word)

Επιστρέφει το μέλος της λίστας που κατέχει θέση που δηλώνει ο αριθμός number ή τον χαρακτήρα όταν το δεύτερο όρισμα είναι μια λέξη.

```
print(item(4 "Cruislet))
print(item(4 [1 2 3 5 6]))
```

- *i*
- *5*

islist(list ή word ή number)

Επιστρέφει true αν το όρισμα είναι λίστα. Σε κάθε άλλη περίπτωση επιστρέφει false.

```
print(islist("Cruislet))
print(islist([1 2 3 5 6]))
```

- *false*
- *true*

Isarray(expr)
Επιστρέφει true αν το όρισμα είναι πίνακας (array). Σε κάθε άλλη περίπτωση επιστρέφει false.

```
make "a array(4)
print(isarray(:a))
```

- *true*

fput(number list), fput(word list), fput(list list)
Δημιουργεί μια νέα λίστα, στην οποία ως πρώτο μέλος τίθεται το πρώτο όρισμα (expr) και έπεται η λίστα που εισάγεται ως δεύτερο όρισμα (list).

```
make "lst ["a 1]
print(fput(12 :lst))
print(fput("b :lst))
print(fput([2 3] :lst))
```

- *[12, a, 1]*
- *[b, a, 1]*
- *[[2, 3], a, 1]*

lput(number list), lput(word list), lput(list list)
Δημιουργεί μια νέα λίστα, η οποία αποτελείται από την λίστα (list), που εισάγεται ως δεύτερο όρισμα, και ως τελευταίο στοιχείο έχει αυτό που εισάγεται με το πρώτο όρισμα.

```
make "lst ["a 1]
print(lput(12 :lst))
print(lput("b :lst))
print(lput([2 3] :lst))
```

- *[a, 1, 12]*
- *[a, 1, b]*
- *[a, 1, [2, 3]]*

itemput(number wordOrlist1 wordOrlist2)
Επιστρέφει την λίστα wordOrlist1 έχεντας αντικαταστήσει το μέλος που δηλώνει ο number με wordOrlist2.

```
print(itemput(2 "LO "OGO))
```

- *LOGO*

```
print(itemput(3 [1 2 3 4] [3 4 5]))
```

- *[1, 2, [3, 4, 5], 4]*

Πίνακες (Array)

Ένας πίνακας (array) στην logo του Cruislet είναι μονοδιάστατος. Το μήκος του πίνακα είναι το πλήθος των κελιών και καθορίζεται από την στιγμή της κατασκευής του. Τους πίνακες τους διαχειριζόμαστε με τις παρακάτω πρωτογενείς διαδικασίες.

array(number)

Δημουργεί έναν μονοδιάστατο πίνακα με πλήθος κελιών ίσο με number. Η παρακάτω εντολή δημιουργεί έναν πίνακα με όνομα "b" και με πλήθος κελιών ίσο με 10.

```
make "b array(10)
```

arraycount(array)

Επιστρέφει το πλήθος των κελιών του μονοδιάστατου πίνακα array.

```
print(arraycount(:b))
```

- *10*

arrayput(array number object)

Θέτει στον πίνακα array και στην θέση number την τιμή object. Στο παρακάτω παράδειγμα ο πίνακας "b" γεμίζει με τις τιμές από το ένα ως το δέκα. Οι θέσεις τους πίνακα αριθμούνται από το μηδέν μέχρι το N-1 όπου N είναι ο αριθμός των κελιών.

```
repeat 10 [
make "b arrayput(:b repcount-1 repcount-1)]
```

arrayget(array number)

Επιστρέφει εκείνο το μέλος του πίνακα array, το οποίο βρίσκεται στην θέση number. Το παρακάτω παράδειγμα τυπώνει στην οθόνη τα περιεχόμενα των κελιών του πίνακα "b".

```
make "i 0
while (:i<arraycount(:b)) [
type(arrayget(:b :i))
make "i :i+1
```

- *0123456789*

Εντολές ελέγχου ροής προγράμματος

#include(filename)

Δίνεται η οδηγία να χρησιμοποιηθεί το αρχείο που δηλώνεται με το όρισμα. Με τον τρόπο αυτό συμπεριλαμβάνουμε αρχεία βιβλιοθήκης.

```
#include "|filename.lgo|
```

Με τη χρήση της παραπάνω εντολής μπορούν να εκτελεστούν οι διαδικασίες που υπάρχουν μέσα στο αρχείο Logo με το όνομα filename.

to <ProcedureName> <Param1> <Param2>

...

end

Με τις δεσμευμένες λέξεις to και end ορίζεται ένα μποκ εντολών της logo, μια διαδικασία. Με τα ορίσματα, που ακολουθούν το όνομα της διαδικασίας, εισάγουμε παραμέτρους στις διαδικασίες.

```
to sum :a :b ;ορίζεται η διαδικασία
print(:a+:b)
end
sum(3 4) ;καλείται να εκτελεστεί η διαδικασία
```

- *7*

if (συνθήκη) [Εντολές]

Είναι η βασική δομή επιλογής. Αν η συνθήκη είναι αληθής εκτελείται το σύνολο των εντολών μέσα στις αγκύλες. Η συνθήκη, απλή ή σύνθετη, πρέπει όταν υπολογίζεται να έχει τιμή «αληθής» ή «ψευδής».

ifelse (συνθήκη) [Εντολές 1][Εντολές 2]

Εναλλακτική σύνταξη της δομής επίλογής if. Αν η συνθήκη είναι αληθής εκτελείται η σειρά των εντολών μέσα στο πρώτο ζευγάρι των αγκυλών, ενώ αν η συνθήκη είναι ψευδής εκτελείται το σύνολο των εντολών στο δεύτερο ζευγάρι αγκυλών.

output(λέξη ή λίστα ή αριθμός)

Σταματά την εκτέλεση της διαδικασίας και επιστρέφει την λίστα ή την λέξη που έχει ως όρισμα.

```
make "mes "|Έφτασα στο εκατό|
make "i 1
to count100
while (true) [
print(:i)
if (:i=100) [output(:mes)]
make "i :i+1
]
end
print(count100)
```

- *..*
- *97*
- *98*
- *99*
- *100*
- *Έφτασα στο εκατό*

repeat

Δομή επανάληψης με την οποία μια λίστα από εντολές εκτελείται, προκαθορισμένα, επαναληπτικά. Συντάσσεται ως εξής:
Repeat number [Εντολές]
Το σύνολο των εντολών, μέσα στις αγκύλες, εκτελούνται τόσες φορές όσες δηλώνει ο «number».

repcount

Έχει την τιμή της τρέχουσας επανάληψης όταν τίθεται μέσα σε ένα βρόχο repeat.
Η εκτέλεση της εντολής που ακολουθεί έχει ως αποτέλεσμα την εμφάνιση στο παράθυρο «έξοδος μηνυμάτων», στην ίδια σειρά, των αριθμών 1,2…10.

```
Repeat 10 [type(repcount)]
```

- *1234567890*

stop

Σταματά την εκτέλεση της διαδικασίας που εκτελείται.

until

Δομή επανάληψης. Η σύνταξή της είναι:
until expr [εντολές]
Η λίστα των εντολών εκτελείται μέχρις ότου η expr πάρει την τιμή true. Επειδή αρχικά υπολογίζεται η τιμή της expr, η οποία πρέπει να έχει τιμή true ή false, είναι πιθανόν να μην εκτελεστούν καμία φορά οι «εντολές».

```
make "a 1
until :a>10 [type(:a)
make "a :a+1]
```

- *1234567890*

while

Δομή επανάληψης. Συντάσσεται ως εξής:
while expr [εντολές]
Η expr πρέπει να έχει, μετά τον υπολογισμό της, τιμή true ή false. Η λίστα των εντολών εκτελείται όσο η expr έχει την τιμή true.

```
make "a 1
while :a<=10 [print(:a)
make "a :a+1]
```

- *1234567890*

wait(number)

Η εκτέλεσή της προκαλεί την παύση της εκτέλεσης του προγράμματος κατά τόσα χιλιοστά του δευτερολέπτου, όσα δηλώνει το όρισμα.

do.until [εντολές] (συνθήκη)

Δομή επανάληψης κατά την οποία ένα σύνολο από εντολές εκτελούνται μέχρις ότου η συνθήκη ελέγχου καταστεί αληθής. Στο παρακάτω παράδειγμα εκτελούνται οι εντολές μέσα στις αγκύλες μέχρις ότου η τιμή της μεταβλητής «a» γίνει μεγαλύτερη από 40.

```
make "a 34
do.until [print(:a) make "a :a+1] (:a>40)
```

- *34*
- *35*
- *36*
- *37*

- *38*
- *39*
- *40*

do.while [εντολές] (συνθήκη)

Δομή επανάληψης κατά την οποία ένα σύνολο από εντολές εκτελούνται ενώ η συνθήκη ελέγχου παραμένει αληθής. Στο παρακάτω παράδειγμα εκτελούνται οι εντολές μέσα στις αγκύλες όσο η τιμή της μεταβλητής «a» είναι μικρότερη από 10.

```
make "a 1
do.while [type(:a) make "a :a+1] (:a<10)
```

- *123456789*

Εντολές Εισόδου– Εξόδου

print (expr)

Στέλνει στο παράθυρο «Έξοδος μηνυμάτων» ένα αντίγραφο του ορίσματος με την μορφή συμβολοσειράς. Αν το όρισμα είναι μια λίστα τυπώνεται με τις περιβάλλουσες αγκύλες. Αν τα όρισμα αποτελείται από δύο ή περισσότερα τμήματα, αυτά τυπώνονται με την σειρά, στη ίδια γραμμή, χωρισμένα με κενά. Στο τέλος στέλνει ένα τέλος γραμμής.

type(expr)

Έχει την ίδια λειτουργία με το print με την διαφορά ότι δεν στέλνει «τέλος γραμμής». Αυτό έχει ως επακόλουθο δύο διαδοχικές εντολές type να τυπώνουν, όταν εκτελούνται, τα ορίσματά τους στην ίδια γραμμή.

Μαθηματικές και λογικές συναρτήσεις

integer(number)

Επιστρέφει το ακέραιο μέρος του ορίσματος.

```
print(integer(5.6))
```

- *5*

int(number)

Παρόμοια με τη integer.

chr(number)

Επιστρέφει τον χαρακτήρα που αντιστοιχεί στο όρισμα, σύμφωνα με τον ASCII κώδικα.

```
print(integer(5.6))
```

- *A*

abs(number)

Επιστρέφει την απόλυτη τιμή του ορίσματος.

```
print(abs(-2))
print(abs(2))
```

- *2*
- *2*

sin(number)

Επιστρέφει την τιμή του ημιτόνου της γωνίας, τόσων μοιρών όσων δηλώνει ο number.

```
print(sin(90))
```

- *1.0*

cos(number)

Επιστρέφει την τιμή του συνημιτόνου της γωνίας, τόσων μοιρών όσων δηλώνει ο number.

```
print(cos(60))
```

- *0.5000000000000001*

tan(number)

Επιστρέφει την τιμή της εφαπτομένης της γωνίας, τόσων μοιρών όσων δηλώνει ο number.

```
print(tan(45))
```

- *0.9999999999999999*

arcsin(number)

Επιστρέφει την τιμή της γωνίας σε μοίρες, της οποίας το ημίτονο είναι ίσο με τον αριθμό που δηλώνεται με τον number.

```
print(arcsin(1))
```

- *90.0*

arcos(number)

Επιστρέφει την τιμή της γωνίας σε μοίρες, της οποίας το συνημίτονο είναι ίσο με τον αριθμό που δηλώνεται με τον number.

```
print(arccos(1))
```

- *0.0*

arctan(number)

Επιστρέφει την τιμή της γωνίας σε μοίρες, της οποίας η εφαπρομένη είναι ίση με τον αριθμό που δηλώνεται με τον number.

```
print(arctan(1))
```

- *45.0*

arctan2(number_y number_x)

Επιστρέφει την κλίση της ευθείας που ορίζεται από τα σημεία (number_x, number_y) και (1,0), και του άξονα x0x'. Η επιστρεφόμενη τιμή είναι μεταξύ του -180 και το 180. Εξορισμού arctan2(0 0) είναι ίση με 0.

```
print(arctan2(1 1))
```

- *45.0*

not(Boolean)

Επιστρέφει την άρνηση της τιμής που εισάγεται ως όρισμα.

```
print(not(false))
```

- *true*

xor(boolean1 boolean2)

Επιστρέφει την τιμή της αποκλειστικής διάζευξης των ορισμάτων. Η επιστρεφόμενη τιμή είναι Boolean, δηλαδή True ή False.

```
print(xor(false true))
```

- *true*

and(boolean1 boolean2)

Επιστέφει την τιμή της λογικής σύξευξης των ορισμάτων. Η επιστρεφόμενη τιμή είναι Boolean δηλαδή True ή False.

```
print(and(false true))
```

- *false*

or(boolean1 boolean2)

Επιστρέφει την τιμή της λογικής διάζευξης των ορισμάτων. Η επιστρεφόμενη τιμή είναι Boolean δηλαδή True ή False.

```
print(or(false true))
```

- *true*

allof(boolean1 boolean2 …)

Επιστρέφει true μόνο αν όλα τα ορίσματα είναι true και false σε κάθε άλλη περίπτωση. Εναλλακτικά το false και το true μπορεί να είναι 0 και 1.

```
print(allof(1 1 1 1 1))
print(allof(true true true false))
```

- *true*
- *false*

anyof(boolean1 boolean2 …)

Επιστρέφει false όταν όλα τα ορίσματα είναι false. Σε κάθε άλλη περίπτωση επιστρέφει true.

```
print(anyof(1 1 1 1 1))
print(anyof(true true true false))
```

- *true*
- *true*

round(number)
Επιστρέφει τον πλησιέστερο ακέραιο αριθμό.

radsin(number)
Επιστρέφει την τιμή του ημιτόνου του ορίσματος, το οποίο εκφράζει γωνία μετρημένη σε ακτίνια (rad).

```
print(radsin(pi/6))
```

- *0.49999999999999994*

radcos (number)
Επιστρέφει την τιμή του συνημιτόνου του ορίσματος, το οποίο εκφράζει γωνία μετρημένη σε ακτίνια (rad).

```
print(radcos(pi/6))
```

- *0.8660254037844387*

radtan(number)
Επιστρέφει την τιμή της εφαπτομένης του ορίσματος, το οποίο εκφράζει γωνία μετρημένη σε ακτίνια (rad).

```
print(radtan(pi/6))
```

- *0.5773502691896257*

radarcsin(number)
Επιστρέφει την τιμή της γωνίας, μετρημένης σε ακτίνια (rad), το ημίτονο της οποίας είναι ίσο με το όρισμα.

```
print(radarcsin(radsin(Pi/6))*(180/Pi))
```

- *29.999999999999996*

radarccos(number)
Επιστρέφει την τιμή της γωνίας, μετρημένης σε ακτίνια (rad), το συνημίτονο της οποίας είναι ίσο με το όρισμα.

```
print(radarccos(radcos(Pi/3))*(180/Pi))
```

- *59.99999999999999*

radarctan(number)
Επιστρέφει την τιμή της γωνίας, μετρημένης σε ακτίνια (rad), η εφαπτομένη της οποίας είναι ίση με το όρισμα.

```
print(radarctan(radtan(Pi/2))*(180/Pi))
```

- *90.0*

sqrt(number)
Επιστρέφει την τετραγωνικήρίζα του ορίσματος.

```
print(sqrt(1024))
```

- *32.0*

pi()
Επιστρέφει την τιμή του αριθμού π.

minus(number)
Επιστρέφει τον αντίθετο του ορίσματος.

```
print(minus(23))
```

- *-23*

sum(number1 number2 …)
Επιστρέφει το άθροισμα των ορισμάτων.

```
print(sum(5 3))
```

- *8*

difference(number1 number2)
Επιστρέφει την διαφορά των δύο ορισμάτων.

```
print(difference(5 3))
```

- *2*

product(number1 number2 …)
Επιστρέφει το γινόμενο των ορισμάτων.

```
print(product(5 3))
```

- *15*

quotient(number1 number2)
Επιστρέφει το πηλίκο των δύο ορισμάτων.

```
print(quotient(5 3))
```

- *1.6666666666666667*

remainder(number1 number2)
Επιστρέφει το υπόλοιπο της ακέραιας διαίρεσης των ορισμάτων.

```
print(remainder(5 3))
```

- *2*

power(number1 number2)
Επιστρέφει το αποτέλεσμα της δύναμης με βάση τον number1 και εκθέτη τον number2.

```
print(power(5 3))
```

- *125.0*

exp(number)
Επιστρέφει την δύναμη με βάση το e και εκθέτη τον number.

```
print(exp(5))
```

- *148.4131591025766*

log(number)
Επιστρέφει τον Νεπέριο λογάριθμο του number.

```
print(log(5))
print(log(exp(5)))
```

- *1.6094379124341003*
- *5.0*

random(number)
Επιστρέφει, τυχαία επιλεγμένο, έναν ακέραιο αριθμό μικρότερο από το όρισμα.

randomize(), randomize(number)
Επανακαθορίζει την αρχική τιμή με βάση την οποία δημιουργείται η σειρά των τυχαίων αριθμών, του οποίους επιστρέφει η random. Με την δεύτερη σύνταξη ως αρχική τιμή τίθεται το όρισμα.

Συναρτήσεις χειρισμού δυαδικών αριθμών

Οι συναρτήσεις της κατηγορίας αυτής δέχονται ως ορίσματα δεκαδικούς αριθμούς. Αυτοί μετατρέπονται σε δυαδικούς, επί των οποίων εκτελείται συγκεκριμένη πράξη, το αποτέλεσμα μετατρέπεται σε δεκαδικό και αυτός επιστρέφεται.

bitnot(number)
Η λειτουργία της είναι να μετατρέπει το όρισμα σε δυαδικό αριθμό, αντιστρέφει τα δυαδικά του ψηφία (το «0» γίνεται «1» και το «1» γίνεται «0), μετατρέπει το αποτέλεσμα σε δεκαδικό αριθμό και αυτόν επιστρέφει.

```
print(bitnot(10))
```

- *-11*

bitand(number1 number2)
Επιστρέφει το αποτέλεσμα του λογικής σύζευξης, στα αντίστοιχα ψηφία της δυαδικής αναπαράστασης, των δύο ορισμάτων.

```
print(bitand(10 3))
```

- *2*

bitor(number1 number2)
Επιστρέφει το αποτέλεσμα του λογικής διάζευξης, στα αντίστοιχα ψηφία της δυαδικής αναπαράστασης, των δύο ορισμάτων.

```
print(bitor(3 5))
```

- *7*

bitxor(number1 number2)
Επιστρέφει το αποτέλεσμα του λογικής αποκλειστικής διάζευξης, στα αντίστοιχα ψηφία της δυαδικής αναπαράστασης, των δύο ορισμάτων.

```
print(bitxor(3 5))
```

- *6*

lshift(number1 number2)

Επιστρέφει το αποτέλεσμα της ολίσθησης, της δυαδικής αναπαράστασης του πρώτου ορίσματος, κατά τόσες θέσεις αριστερά, όσες δηλώνει το δεύτερο όρισμα. Αν το δεύτερο όρισμα είναι αρνητικός αριθός, τότε η ολίσθηση γίνεται προς τα δεξιά.

```
print(lshift(4 1))
print(lshift(4 -1))
```

- *8*
- *2*

ashift(number1 number2)

Η λειτουργία της είναι ίδια με αυτή της lshift, μόνο που κατά την ολίσθηση διατηρείται το πρόσημο του πρώτου ορίσματος. Το πρόσημο ενός δυαδικού αριθμού είναι θετικό, αν το περισσότερο σημαντικό ψηφίο είναι 0 και αρνητικό αν αυτό είναι 1.

```
print(lshift(-15 -2))
print(ashift(-15 -2))
```

- *1073741820*
- *-4*

Περιεχόμενα