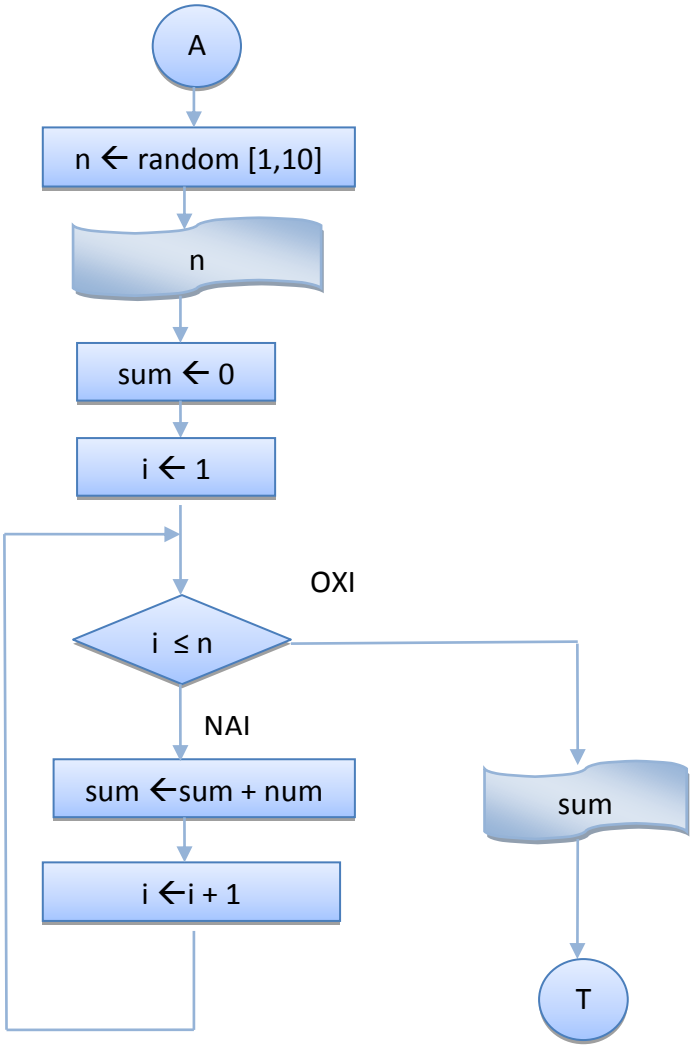
4 ΕΝΤΟΛΕΣ ΕΠΑΝΑΛΗΨΗΣ - for

Υπάρχουν προβλήματα, στα οποία ο αριθμός των επαναλήψεων κάποιων εντολών είναι **γνωστός** εκ των προτέρων, όπως στο επόμενο παράδειγμα :

4. 1 Πρόγραμμα για τον Υπολογισμό του Αθροίσματος 1+2+… + n με την Εντολή Επανάληψης while

Να γραφεί Αλγόριθμος/πρόγραμμα, το οποίο θα δημιουργεί έναν τυχαίο ακέραιο αριθμό μεταξύ του `1` και `10`, ο οποίος θα αποθηκεύεται στη μεταβλητή **n**, θα εμφανίζει την τιμή του και θα υπολογίζει και θα εμφανίζει το άθροισμα 1 + 2 + ... + **n**.

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



ΑΛΓΟΡΙΘΜΟΣ

1. Δημιουργώ έναν τυχαίο ακέραιο αριθμό `n` στο `[1, 10]`
2. Τον εμφανίζω
3. Δίνω αρχική τιμή το μηδέν στον αθροιστή `sum` (`sum` ← 0)
4. Δίνω αρχική τιμή το `1` στον αριθμό `i` (`i ← 1`)
5. **Για όσο** ο αριθμός `i` είναι ≤ n (`i ≤ n`)
   a. Προσθέτω το `i` στο `sum` (`sum ← sum + i`)
   b. Αυξάνω την τιμή του αριθμού κατά `1` (`i ← i + 1`)
6. Εμφανίζω την τιμή του αθροιστή `sum`

ΠΡΟΓΡΑΜΜΑ

```
public class WhileFor12n {
/*Πρόγραμμα, το οποίο  δημιουργεί έναν τυχαίο ακέραιο αριθμό μεταξύ του
1 και 10, ο οποίος θα αποθηκεύεται στη μεταβλητή n,  εμφανίζει την τιμή
του και  υπολογίζει και  εμφανίζει το άθροισμα 1 + 2 + ... + n.*/
    public static void main(String[] args) {
       int n, sum, i;

       // Δημιουργία - Εμφάνιση τυχαίου ακέραιου αριθμού στο 1 - 10
        n = (int) ( Math.random()*10 + 1);
        System.out.println("n = " + n );

        // Αρχικές τιμές στο άθροισμα και τον αριθμό
        sum = 0;
        i = 1;

        // Για όσο ο αριθμός δεν ξεπέρασε το n
        while (i <= n)
        {
          // Πρόσθεση του αριθμού στον αθροιστή sum
          sum += i;

          // Αύξηση του αριθμού i κατά 1
          i++;
        }

       // Εμφάνιση αθροίσματος sum
       System.out.println("Το άθροισμά 1+2+...+" + n + " είναι " + sum +
"\n\n");
    }
}
```

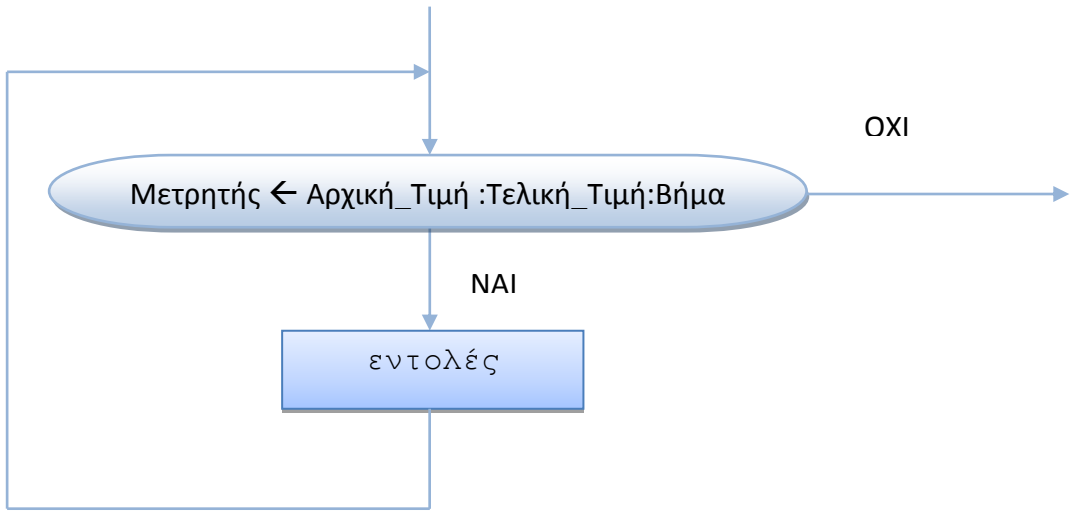
Έξοδος Προγράμματος

```
run:
n = 6
Το άθροισμα 1+2+...+6 είναι 21
BUILD SUCCESSFUL (total time: 0 seconds)
```

ΑΣΚΗΣΗ 4.1 : Να τροποποιηθεί ο προηγούμενος αλγόριθμος ώστε να υπολογίζει/ εμφανίζει και το **Μέσο Όρο** των αριθμών `1 + 2 + ... + n` με **do…while**.

4. 2 Η Εντολή Επανάληψης for

Στα προβλήματα με γνωστό αριθμό επαναλήψεων, οι εντολές ανάθεσης **αρχικής τιμής** στο μετρητή, ελέγχου της **συνθήκης**, αν **ο μετρητής ξεπέρασε την τελική τιμή** και **ενημέρωσης της τιμής του μετρητή** μέσα στο σώμα της επανάληψης μπορούν να συμπεριληφθούν σε μια εντολή, την εντολή `for`, της οποίας η σύνταξη (σε μορφή διαγράμματος ροής, αλγορίθμου και `Java`) είναι :



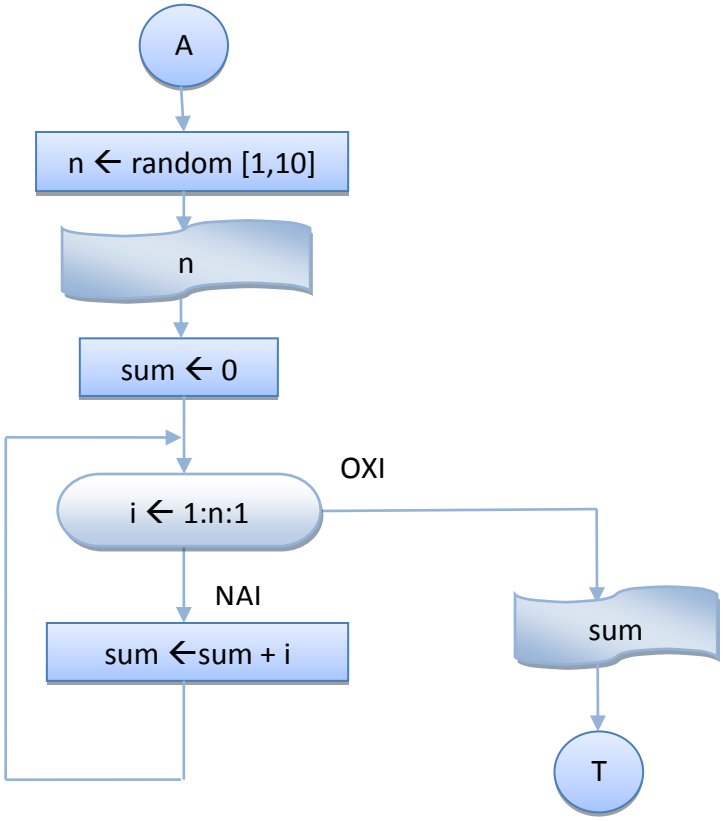
Για <μετρητής>=<αρχική τιμή>:<τελική τιμή>:<βήμα>
εντολές;

```
for (<αρχική τιμή μετρητή>; <ο μετρητής ξεπέρασε την τελική τιμή?>; <ενημέρωση
     τιμής μετρητή>)
     {
        εντολές;
     }
```

4.2.1 Πρόγραμμα για τον Υπολογισμό του Αθροίσματος 1+2+…+ n με την Εντολή Επανάληψης for

Να γραφεί Αλγόριθμος/πρόγραμμα, το οποίο θα δημιουργεί έναν τυχαίο ακέραιο αριθμό μεταξύ του `1` και `10`, ο οποίος θα αποθηκεύεται στη μεταβλητή **n**, θα εμφανίζει την τιμή του και θα υπολογίζει και θα εμφανίζει το άθροισμα `1 + 2 + ... + n` **με την εντολή επανάληψης for**.

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



ΑΛΓΟΡΙΘΜΟΣ

1. Δημιουργώ έναν τυχαίο ακέραιο αριθμό `n` στο `[1, 10]`
2. Τον εμφανίζω
3. Δίνω αρχική τιμή το μηδέν στον αθροιστή `sum` (`sum` ← 0)
4. **Για** τις τιμές του μετρητή `i` από το `1` μέχρι και το `n` με βήμα `1`
   Προσθέτω το `i` στο `sum` (`sum ← sum + i`)
5. Εμφανίζω την τιμή του αθροιστή `sum`

ΠΡΟΓΡΑΜΜΑ

```
public class ForSum {
/*Πρόγραμμα, το οποίο  δημιουργεί έναν τυχαίο ακέραιο αριθμό μεταξύ του
1 και 10, ο οποίος αποθηκεύεται στη μεταβλητή n,  εμφανίζει την τιμή
του και υπολογίζει και εμφανίζει το άθροισμα 1 + 2 + ... + n με την
εντολή for.
*/
    public static void main(String[] args) {

        int n, sum, i;

       // Δημιουργία - Εμφάνιση τυχαίου ακέραιου αριθμού στο 1 - 10
        n = (int) ( Math.random()*10 + 1);
        System.out.println("n = " + n );

        // Αρχική τιμή 0 στο άθροισμα
        sum = 0;

        // Για την τιμή του μετρητή i από το 1 μέχρι και το n
        for ( i = 1;i <= n;i++ )
          // Πρόσθεση του αριθμού στον αθροιστή sum
          sum += i;

        // Εμφάνιση αθροίσματος sum
        System.out.println("Το άθροισμα 1+2+...+" + n + " είναι " + sum
               + "\n");
    }
}
```

Έξοδος Προγράμματος

```
run:
n = 6
Το άθροισμα 1+2+...+6 είναι 21
BUILD SUCCESSFUL (total time: 0 seconds)
```

ΑΣΚΗΣΗ 4.2 : Να τροποποιηθεί ο προηγούμενος αλγόριθμος ώστε να υπολογίζει και να εμφανίζει και το **Μέσο Όρο** των αριθμών `1 + 2 + ... + n`.

ΑΣΚΗΣΗ 4.3 : Να τροποποιηθεί ο προηγούμενος αλγόριθμος ώστε να υπολογίζει και να εμφανίζει το **Άθροισμα** και το **Μέσο Όρο** των αριθμών `n+(n−1)+…+2+1` (ο μετρητής θα αρχίσει από το `n` και θα **μειώνεται** κατά `1`, μέχρι και το `1`).

Παρατηρήσεις

Η εντολή `for` δεν είναι απαραίτητο να περιέχει και την <αρχική τιμή στο μετρητή> τον έλεγχο αν <ο μετρητής ξεπέρασε την τελική τιμή> και την <ενημέρωση της τιμής του μετρητή>.

Παράδειγμα

Από την προηγούμενη πλήρη εντολή επανάληψης `for`

```
for (i = 1;i <= n;i++)
  sum += i;
```

μπορεί να λείπουν κάποια απ' αυτά ή όλα, αρκεί να υπάρχουν τα αντίστοιχα ερωτηματικά. Π.χ. θα μπορούσε να λείπει :

- Η <αρχική τιμή στο μετρητή>, οπότε θα έχουμε :

```
i = 1;
for (;i <= n;i++)
  sum += i;
```

- Η <ενημέρωση της τιμής του μετρητή>, οπότε θα έχουμε:

```
for (i = 1;i <= n;){
  sum += i;
  i++;
}
```

- Η <αρχική τιμή στο μετρητή> και η <ενημέρωση της τιμής του μετρητή>, οπότε θα έχουμε :

```
i = 1;
for (;i <= n;){
  sum += i;
  i++;
}
```

- Το σώμα της εντολής επανάληψης . Π.χ.

```
for (i = 1;i <= n; sum += i++);
```

- Όλα τα παραπάνω και ο έλεγχος αν <ο μετρητής ξεπέρασε την τελική τιμή> (ατέρμονος βρόχος), οπότε απαιτείται η χρήση της εντολής break.

Παράδειγμα

Με τις επόμενες εντολές διαβάζουμε χαρακτήρες μέχρι να δώσουμε το χαρακτήρα q (quit).

```
char ch;
for (;;){
   ch = (char) System.in.read(); // Εισαγωγή χαρακτήρα
   if ( ch = 'q' ) break; // Τερματισμός βρόχου με q = quit
}
```

❖ Στο προηγούμενο παράδειγμα, απαιτείται και η χρήση του throws java.io.IOException στη δήλωση της μεθόδου που χρησιμοποιεί το προηγούμενο τμήμα του κώδικα.

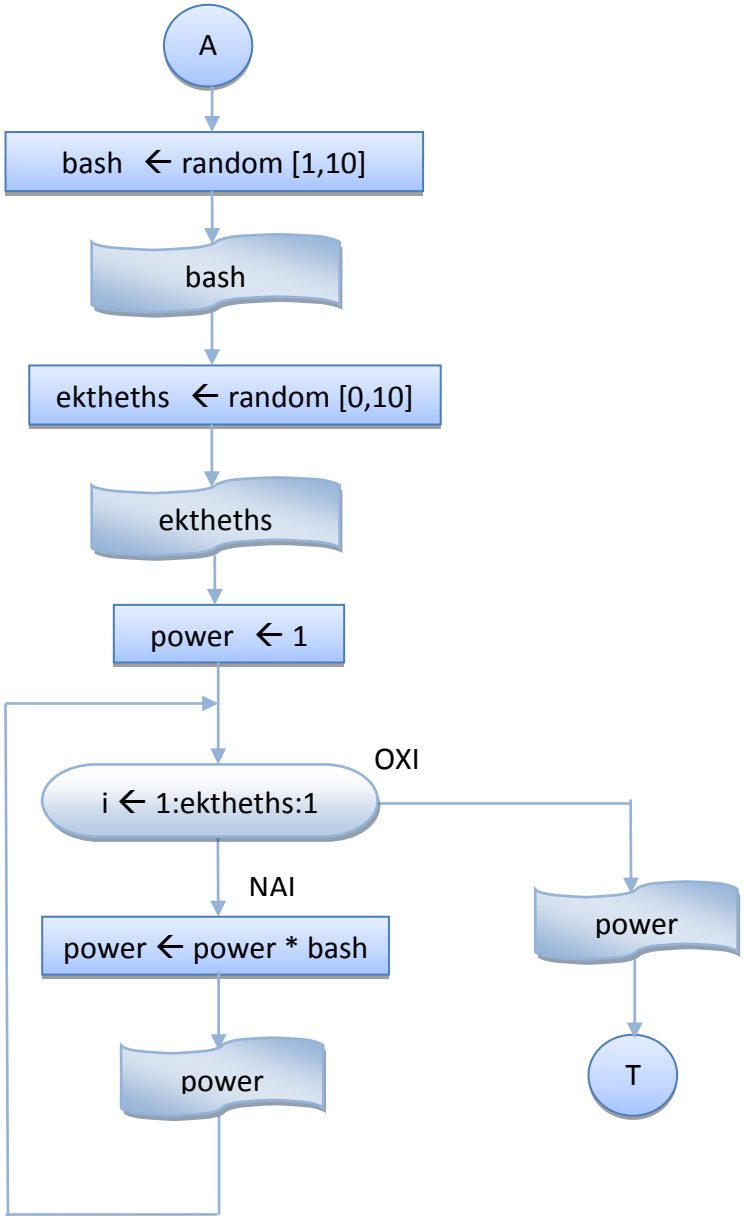
4.2.2 Πρόγραμμα για τον Υπολογισμό του x^y με την Εντολή Επανάληψης for

Να γραφεί Αλγόριθμος/πρόγραμμα, το οποίο θα προσομοιώνει τη μέθοδο Math.pow(Βάση, Εκθέτης) με **Βάση** έναν **ακέραιο** αριθμό και **εκθέτη** έναν **ακέραιο** αριθμό. Δημιουργεί έναν τυχαίο **ακέραιο** αριθμό στο 1-10 για τη Βάση και έναν τυχαίο ακέραιο αριθμό στο 0-10 για τον Εκθέτη και εμφανίζει τις τιμές τους. Για να υπολογίσει τη δύναμη Βάση^Εκθέτη, δίνει την τιμή 1 σαν αρχική τιμή στη δύναμη και με την εντολή for πολλαπλασιάζει τη δύναμη με τη Βάση, όσες φορές είναι η ακέραια τιμή του εκθέτη και εμφανίζει κάθε φορά την τιμή της δύναμης.

ΑΛΓΟΡΙΘΜΟΣ

1. Δημιουργώ έναν τυχαίο **ακέραιο** αριθμό `bash` στο `[1, 10]`
2. Τον εμφανίζω
3. Δημιουργώ έναν τυχαίο **ακέραιο** αριθμό `ektheths` στο `[0, 10]`
4. Τον εμφανίζω
5. Δίνω αρχική τιμή στη `Δύναμη power = 1`
6. **Για** τις τιμές του μετρητή `i` από το `1` μέχρι και τον εκθέτη `ektheths`
   Πολλαπλασιάζω τη Δύναμη με τη Βάση (`power ← power * bash`)
   Εμφανίζω την τιμή της Δύναμης `power`
7. Εμφανίζω την τελική τιμή της Δύναμης `power`

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



ΠΡΟΓΡΑΜΜΑ

```
public class PowerFor {
/*
Πρόγραμμα, το οποίο προσομοιώνει τη μέθοδο pow( Βάση, Εκθέτης ) με Βάση
έναν ακέραιο αριθμό και εκθέτη έναν ακέραιο αριθμό. Δημιουργεί έναν
τυχαίο ακέραιο αριθμό στο 1-10 για τη Βάση και έναν τυχαίο ακέραιο
αριθμό στο 0-10 για τον Εκθέτη και εμφανίζει τις τιμές τους. Για να
υπολογίσει τη δύναμη Βάση^Εκθέτη, δίνει την τιμή 1 σαν αρχική τιμή στη
δύναμη και με την εντολή for πολλαπλασιάζει τη δύναμη με τη Βάση, όσες
φορές είναι η ακέραια τιμή του εκθέτη και εμφανίζει κάθε φορά την τιμή
της δύναμης.
*/

public static void main(String[] args) {
       int ektheths, i;
       int power;

       // Δημιουργία τυχαίου ακέραιου αριθμού στο 1 - 10 για τη Βάση
       int bash = (int)(Math.random()*10) + 1;

       // Εμφάνιση της Τιμής της Βάσης
       System.out.println("Η Βάση είναι : " + bash );

       // Δημιουργία τυχαίου ακέραιου αριθμού στο 0 - 10 για τον εκθέτη
       ektheths = (int) ( Math.random()*10);

       // Εμφάνιση της Τιμής του Εκθέτη
       System.out.println("Ο Εκθέτης είναι : " + ektheths );


       // Αρχική Τιμή στη Δύναμη = 1
       power = 1;

       // Για τόσες φορές όσες η τιμή του εκθέτη
       for ( i = 1; i <= ektheths; i++ ) {
           // Υπολογισμός Επόμενης Τιμής της Δύναμης
           power = power * bash;

           // Εμφάνιση της νέας Τιμής της Δύναμης
           System.out.println("Η " + i + "-η Δύναμη είναι : " + power );
       }
       // Εμφάνιση της Τελικής Τιμής της Δύναμης
          System.out.println("\nΤελική Τιμή για τη Δύναμη = " + power );
       }
}
```

Έξοδος Προγράμματος

```
run:
Η Βάση είναι : 6
Ο Εκθέτης είναι : 0

Τελική Τιμή για τη Δύναμη = 1
BUILD SUCCESSFUL (total time: 0 seconds)

run:
Η Βάση είναι : 6
Ο Εκθέτης είναι : 1
Η 1-η Δύναμη είναι : 6

Τελική Τιμή για τη Δύναμη = 6
BUILD SUCCESSFUL (total time: 0 seconds)

run:
Η Βάση είναι : 3
Ο Εκθέτης είναι : 4
Η 1-η Δύναμη είναι : 3
Η 2-η Δύναμη είναι : 9
Η 3-η Δύναμη είναι : 27
Η 4-η Δύναμη είναι : 81

Τελική Τιμή για τη Δύναμη = 81
BUILD SUCCESSFUL (total time: 0 seconds)

run:
Η Βάση είναι : 1
Ο Εκθέτης είναι : 1
Η 1-η Δύναμη είναι : 1

Τελική Τιμή για τη Δύναμη = 1
BUILD SUCCESSFUL (total time: 0 seconds)

run:
Η Βάση είναι : 1
Ο Εκθέτης είναι : 6
Η 1-η Δύναμη είναι : 1
Η 2-η Δύναμη είναι : 1
Η 3-η Δύναμη είναι : 1
Η 4-η Δύναμη είναι : 1
Η 5-η Δύναμη είναι : 1
Η 6-η Δύναμη είναι : 1

Τελική Τιμή για τη Δύναμη = 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

ΑΣΚΗΣΗ 4.4 : Να τροποποιηθεί ο προηγούμενος αλγόριθμος ώστε να ελέγχει και την περίπτωση που **η βάση είναι 0 ή 1** και να εμφανίζει την τιμή της, **χωρίς** να χρειαστεί να κάνει καμιά επανάληψη.

4. 3 Εμφωλευμένες Εντολές Επανάληψης for-while, break, continue

Στα επόμενα παραδείγματα εξετάζεται η χρήση της εντολής επανάληψης `for` μέσα στο σώμα μιας εντολής επανάληψης `while`, η χρήση της εντολής επανάληψης `for` μέσα στο σώμα μιας άλλης εντολής επανάληψης `for`, και η χρήση των εντολών `break` και `continue`.

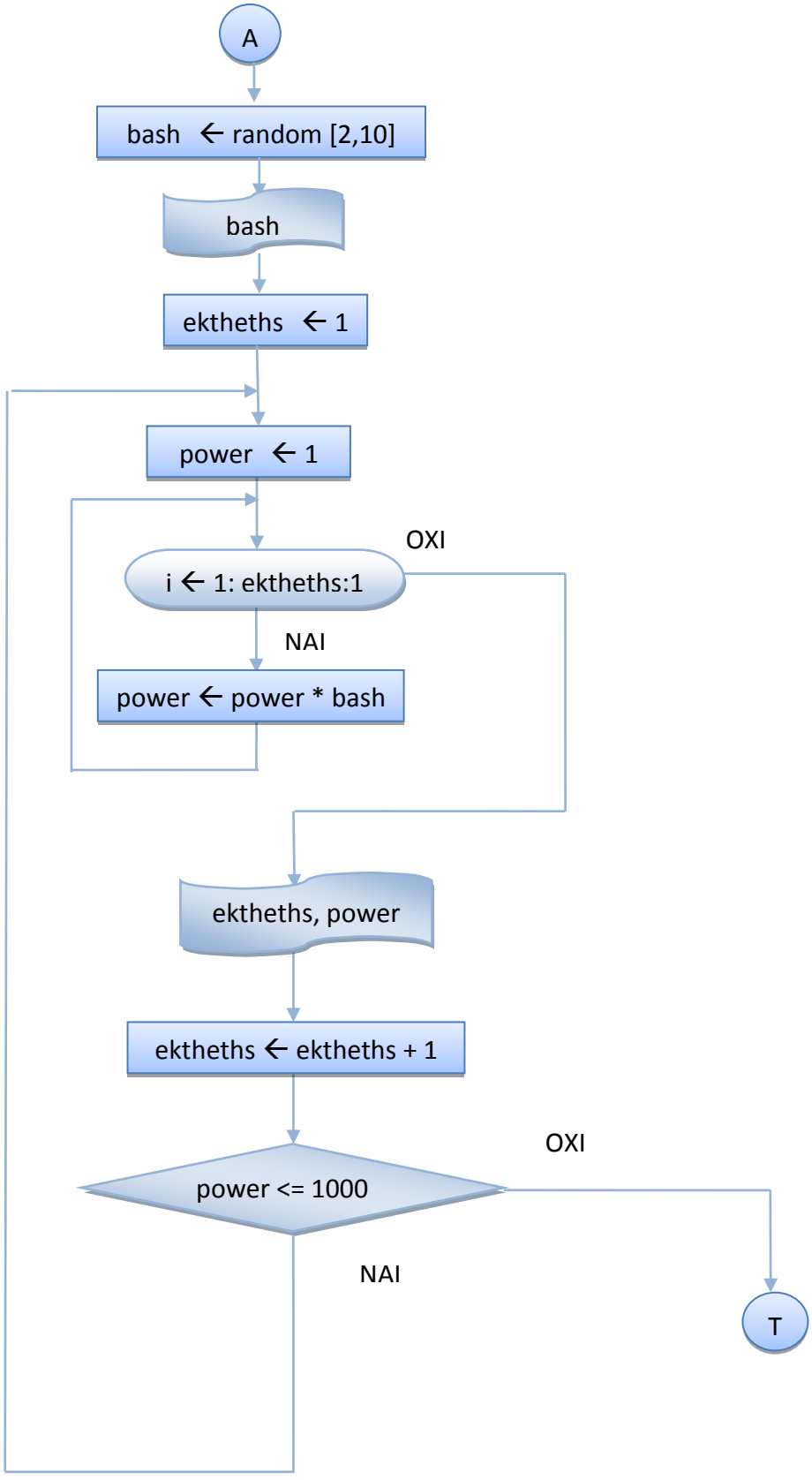
4.3.1 Εμφωλευμένοι Βρόχοι (for – while)

Να γραφεί Αλγόριθμος/πρόγραμμα, το οποίο θα υπολογίζει και θα εμφανίζει όλες τις **Δυνάμεις** ενός τυχαίου ακέραιου αριθμού στο [2,10] μέχρι που η Δύναμη να ξεπεράσει το **1000** χρησιμοποιώντας τις Εντολές Επανάληψης **`for`** για τον υπολογισμό της κάθε δύναμης και **`do while`** για τον έλεγχο του τερματισμού.

ΑΛΓΟΡΙΘΜΟΣ

1. Δημιουργώ έναν τυχαίο **ακέραιο** αριθμό `bash` στο `[2, 10]`
2. Τον εμφανίζω
3. Αρχική Τιμή στον Εκθέτη `ektheths` ← `1`
4. **Κάνε** τα παρακάτω
   Δίνω αρχική τιμή στη `Δύναμη power` ← `1`
   Για τις τιμές του μετρητή `i` από το `1` μέχρι και τον εκθέτη `ektheths`
   Πολλαπλασιάζω τη Δύναμη με τη Βάση (`power` ← `power * bash`)
   Εμφανίζω την τιμή της Δύναμης και του εκθέτη
   Αυξάνω την τιμή του εκθέτη κατά 1
   Για όσο η Δύναμη είναι μικρότερη ή ίση του `1000`

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



ΠΡΟΓΡΑΜΜΑ

```
public class DoWhileFor {
/*
Πρόγραμμα, το οποίο Υπολογίζει και εμφανίζει όλες τις Δυνάμεις ενός
τυχαίου ακέραιου αριθμού στο [2,10] μέχρι που η Δύναμη να ξεπεράσει το
1000 χρησιμοποιώντας τις Εντολές Επανάληψης for για τον υπολογισμό της
κάθε δύναμης και do while για τον έλεγχο του τερματισμού.
*/
public static void main(String[] args) {
        int ektheths, i, power;

       // Δημιουργία τυχαίου ακέραιου αριθμού στο 2 - 10 για τη Βάση
       int bash = (int)(Math.random()*9) + 2;

       // Εμφάνιση της Τιμής της Βάσης
       System.out.println("Η Βάση είναι : " + bash );

       // Αρχική Τιμή στον Εκθέτη = 1
       ektheths = 1;

       do {
         // Αρχική Τιμή στη Δύναμη = 1
         power = 1;

         // Για τόσες φορές όσες η τιμή του εκθέτη
         for ( i = 1; i <= ektheths; i++) {
           // Υπολογισμός Επόμενης Τιμής της Δύναμης
           power = power * bash;
         }
         // Εμφάνιση του εκθέτη και της νέας Τιμής της Δύναμης
                        System.out.println("Η Δύναμη του αριθμού " +
                          bash + "^" + ektheths + " είναι : " + power );

         // Αϋξηση του Εκθέτη κατά 1
         ektheths += 1;
       }
       while (power <= 1000);
}
}
```

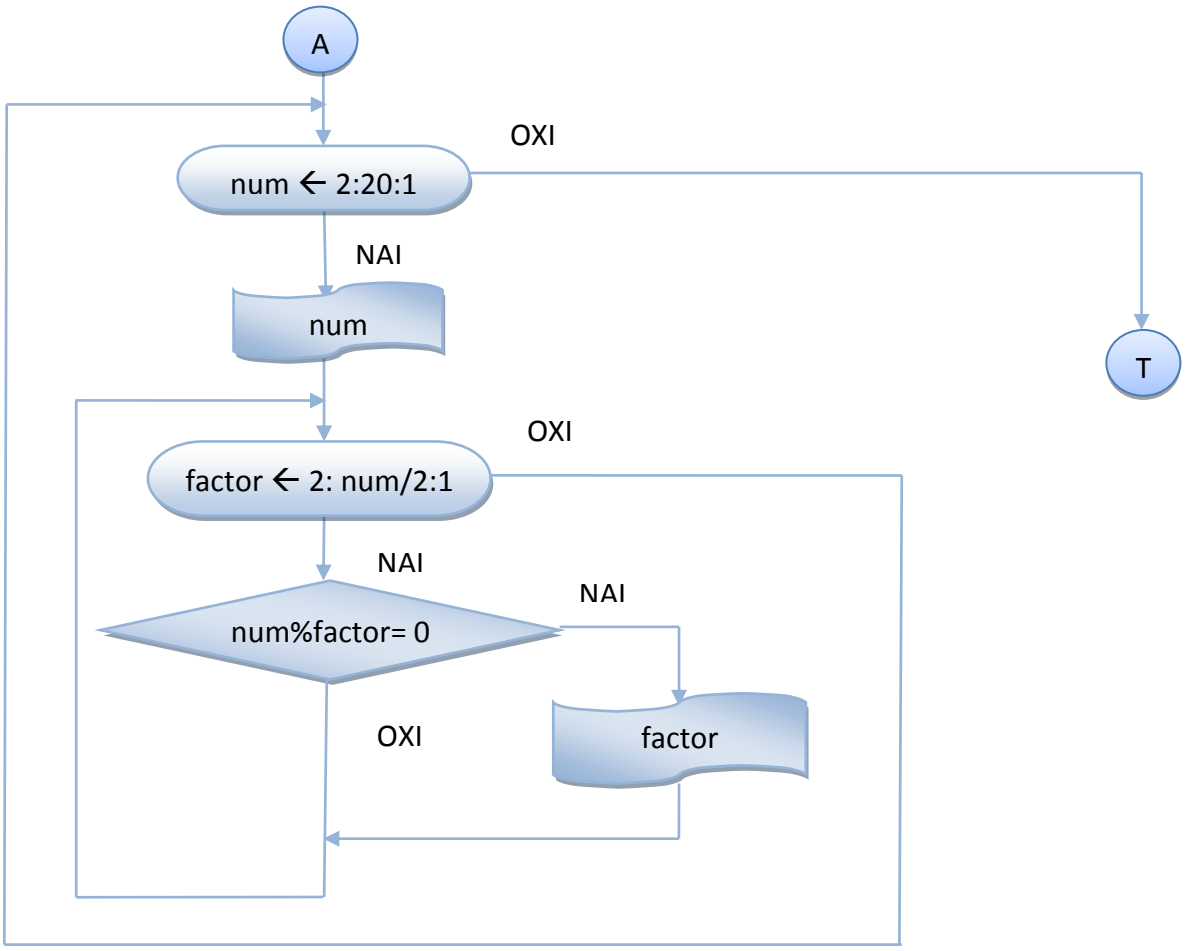
Έξοδος Προγράμματος

```
run:
Η Βάση είναι : 2
Η Δύναμη του αριθμού 2^1 είναι : 2
Η Δύναμη του αριθμού 2^2 είναι : 4
Η Δύναμη του αριθμού 2^3 είναι : 8
Η Δύναμη του αριθμού 2^4 είναι : 16
Η Δύναμη του αριθμού 2^5 είναι : 32
Η Δύναμη του αριθμού 2^6 είναι : 64
Η Δύναμη του αριθμού 2^7 είναι : 128
Η Δύναμη του αριθμού 2^8 είναι : 256
Η Δύναμη του αριθμού 2^9 είναι : 512
Η Δύναμη του αριθμού 2^10 είναι : 1024
BUILD SUCCESSFUL (total time: 0 seconds)
```

4.3.2 Εμφωλευμένοι Βρόχοι (for – for)

Να γραφεί Αλγόριθμος/πρόγραμμα, το οποίο θα υπολογίζει όλους τους **παράγοντες** των αριθμών από το `2` μέχρι το `20` (για τον κάθε αριθμό θα βρίσκει και θα εμφανίζει τους **διαιρέτες** του **εκτός** της μονάδας και του ίδιου του αριθμού) χρησιμοποιώντας **δύο** Εντολές Επανάληψης **`for`**.

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



ΑΛΓΟΡΙΘΜΟΣ

1. **Για** τις τιμές του μετρητή `num` από το `2` μέχρι και το `20`
   Εμφάνιση της τιμής του μετρητή `num`
   Για τις τιμές του `factor` από το `2` μέχρι και το `num/2`
   Αν ο αριθμός `num` διαιρείται ακριβώς με το `factor`
   Εμφανίζω την τιμή του παράγοντα `factor`

ΠΡΟΓΡΑΜΜΑ

```
public class ForFor {
/*Πρόγραμμα, το οποίο υπολογίζει όλους τους παράγοντες των αριθμών από
το 2 μέχρι το 20 ( για τον κάθε αριθμό θα βρίσκει και θα εμφανίζει  τους
διαιρέτες του εκτός της μονάδας και του ίδιου του αριθμού )
χρησιμοποιώντας δύο Εντολές Επανάληψης for.*/
    public static void main(String[] args) {
        int num, factor;

        // Για τις τιμές του μετρητή num από το 2 μέχρι και το 20
        for ( num = 2;num <= 20;num++ ) {

            // Εμφάνιση της τιμής του μετρητή num
            System.out.print("Ο αριθμός " + num + " έχει παράγοντες : "
);

            // Για τις τιμές του factor από το 2 μέχρι και το num/2
            for ( factor = 2;factor <= num/2;factor++ )

                // Αν ο αριθμός num διαιρείται ακριβώς με το factor
                if ( num % factor == 0)

                    // Εμφάνιση της τιμής του παράγοντα factor
                    System.out.print(" " + factor + " " );

            System.out.println();
        }
     }
}
```

Έξοδος Προγράμματος

```
run:
Ο αριθμός 2 έχει παράγοντες :
Ο αριθμός 3 έχει παράγοντες :
Ο αριθμός 4 έχει παράγοντες :  2
Ο αριθμός 5 έχει παράγοντες :
Ο αριθμός 6 έχει παράγοντες :  2  3
Ο αριθμός 7 έχει παράγοντες :
Ο αριθμός 8 έχει παράγοντες :  2  4
Ο αριθμός 9 έχει παράγοντες :  3
Ο αριθμός 10 έχει παράγοντες :  2  5
Ο αριθμός 11 έχει παράγοντες :
Ο αριθμός 12 έχει παράγοντες :  2  3  4  6
Ο αριθμός 13 έχει παράγοντες :
Ο αριθμός 14 έχει παράγοντες :  2  7
Ο αριθμός 15 έχει παράγοντες :  3  5
Ο αριθμός 16 έχει παράγοντες :  2  4  8
Ο αριθμός 17 έχει παράγοντες :
Ο αριθμός 18 έχει παράγοντες :  2  3  6  9
Ο αριθμός 19 έχει παράγοντες :
Ο αριθμός 20 έχει παράγοντες :  2  4  5  10
BUILD SUCCESSFUL (total time: 0 seconds)
```

Άσκηση 4.5 : Να τροποποιηθεί το προηγούμενο πρόγραμμα, ώστε να εμφανίζει **μόνο** τους αριθμούς που έχουν παράγοντες, **ΟΧΙ** και τους πρώτους.

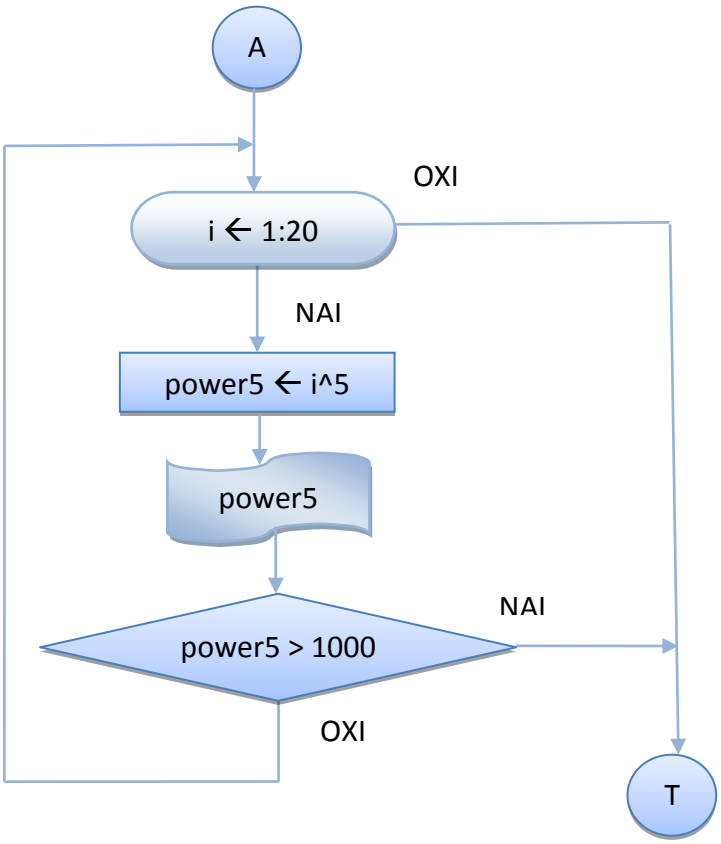
4.3.3 Η Εντολή break

Η εντολή `break` χρησιμοποιείται για τη διακοπή οποιουδήποτε βρόχου επανάληψης, ακόμη κι αν εξακολουθεί να ισχύει η συνθήκη. Εκτός από την έξοδο από κάθε περίπτωση (`case`) της εντολής επιλογής (`switch`) και την έξοδο από τον ατέρμονο βρόχο επανάληψης `for (;;)` μπορεί να χρησιμοποιηθεί για τη διακοπή οποιουδήποτε βρόχου επανάληψης, όπως φαίνεται στο επόμενο παράδειγμα :

Παράδειγμα

Να γραφεί Αλγόριθμος/πρόγραμμα, το οποίο θα υπολογίζει με τη χρήση της μεθόδου `Math.pow()` την **πέμπτη** δύναμη των ακέραιων αριθμών από το `1` μέχρι και το `20` και θα την εμφανίζει. Το πρόγραμμα θα τερματίζει με την εντολή `break`, αν η τιμή της πέμπτης δύναμης κάποιου από τους αριθμούς `1-20` είναι μεγαλύτερη του `1000`.

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



ΑΛΓΟΡΙΘΜΟΣ

Για τους αριθμούς `i = 1:20`
    Υπολόγισε την Πέμπτη δύναμη `power5` του αριθμού `i`
    Εμφάνισε την Πέμπτη δύναμη `power5` του αριθμού `i`
    Αν η `power5` είναι μεγαλύτερη του `1000`
            `Διακοπή Βρόχου Επανάληψης`

ΠΡΟΓΡΑΜΜΑ

```
public class Power5 {
    /*
    Πρόγραμμα το οποίο υπολογίζει με τη χρήση της μεθόδου Math.pow() την
    πέμπτη δύναμη των ακέραιων αριθμών από το 1 μέχρι και το 20 και  την
    εμφανίζει. Το πρόγραμμα  τερματίζει με την εντολή break, αν  η τιμή
    της πέμπτης δύναμης κάποιου από τους αριθμούς 1-20 είναι μεγαλύτερη
    του 1000
     */
    public static void main(String[] args) {
        int i;
        int power5;
        // Για τους αριθμούς από το 1 μέχρι και το 20
        for ( i=1;i<=20;i++ ) {
          // Υπολογισμός i^5
          power5 = (int)Math.pow(i,5);
          // Εμφάνιση i^5
          System.out.println(i + "^5 = " + power5 );
          // Έλεγχος - Διακοπή, αν η δύναμη ξεπέρασε το 1000
          if (power5 > 1000 ) break;
        }
    }
}
```

Έξοδος Προγράμματος

```
run:
1^5 = 1
2^5 = 32
3^5 = 243
4^5 = 1024
BUILD SUCCESSFUL (total time: 0 seconds)
```

Άσκηση 4.6 : Να τροποποιηθεί ο προηγούμενος Αλγόριθμος/πρόγραμμα ώστε να ΜΗΝ εμφανίζει την τιμή του `power5`, αν είναι μεγαλύτερη του `1000`.

Άσκηση 4.7 : Να τροποποιηθεί ο προηγούμενος Αλγόριθμος/πρόγραμμα ώστε να κάνει το ίδιο **ΧΩΡΙΣ** τη χρήση των εντολών `for` και `break`.

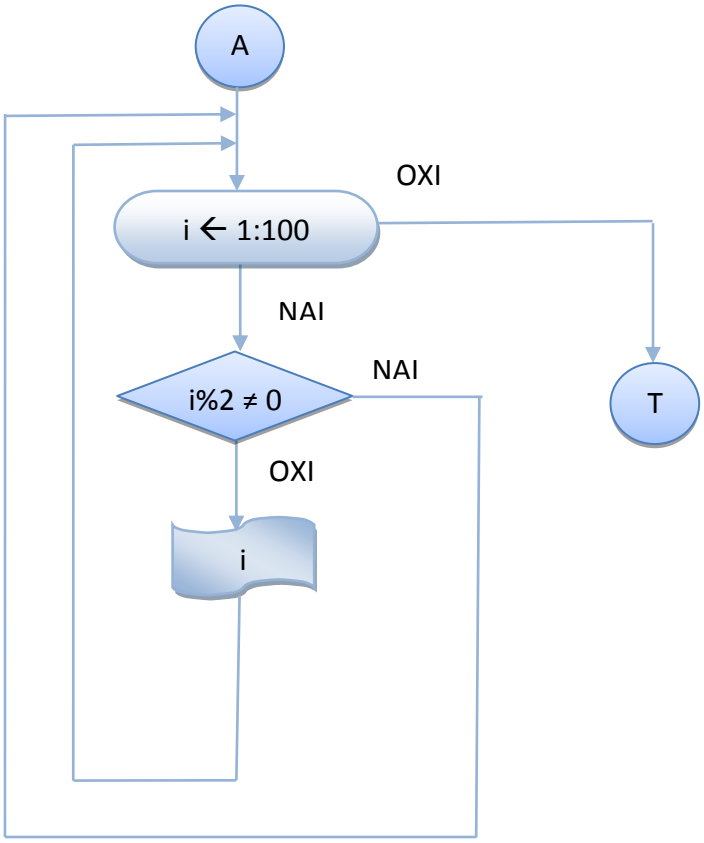
4.3.4 Η Εντολή continue

Η εντολή `continue` είναι το συμπλήρωμα της εντολής `break` και χρησιμοποιείται για να στείλει τον έλεγχο στη συνθήκη του οποιουδήποτε βρόχου επανάληψης, όπως φαίνεται στο επόμενο παράδειγμα :

Παράδειγμα

Να γραφεί Αλγόριθμος/πρόγραμμα, το οποίο θα βρίσκει και θα εμφανίζει όλους τους άρτιους ακέραιους αριθμούς από το 1 ως το 20 με τη χρήση της εντολής `continue`.

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



ΑΛΓΟΡΙΘΜΟΣ

Για τους αριθμούς `i = 1:20`
    Αν ο αριθμός `i` **δεν** διαιρείται ακριβώς με το `2`
        Συνέχισε με τον επόμενο αριθμό
    Εμφάνισε τον αριθμό `i`

ΠΡΟΓΡΑΜΜΑ

```
public class ForContinue {
    /*
     * Πρόγραμμα, το οποίο βρίσκει και  εμφανίζει όλους τους άρτιους
     * ακέραιους αριθμούς από το 1 ως το 20 με τη χρήση της εντολής
     * continue.
     */
    public static void main(String[] args) {
        int i;
        // Για τους αριθμούς από το 1 μέχρι και το 200
        for ( i=1;i<=20;i++ ) {
          // Έλεγχος αν ο αριθμός είναι άρτιος
          if (i%2 != 0) continue;
          // Εμφάνιση i
          System.out.println(i  );
        }
    }
}
```

Έξοδος Προγράμματος

```
run:
2
4
6
8
10
12
14
16
18
20
BUILD SUCCESSFUL (total time: 0 seconds)
```

Άσκηση 4.8 : Να τροποποιηθεί ο προηγούμενος Αλγόριθμος/πρόγραμμα ώστε να κάνει το ίδιο **ΧΩΡΙΣ** τη χρήση της εντολής `continue`.