

ΥΠΟΛΟΓΙΣΤΕΣ ΙΙ

Εντολές επανάληψης

1

Εντολές επανάληψης

Στη C++ υπάρχουν 3 διαφορετικές εντολές επανάληψης:

- **while**
- **for**
- **do-while**

2

ΥΠΟΛΟΓΙΣΤΕΣ ΙΙ

Εντολή *while*

3

Παράδειγμα #1

Κατασκευάστε πρόγραμμα που για δεδομένο ακέραιο N θα εμφανίζει στην οθόνη τους αριθμούς από 1 έως N

4

Παράδειγμα #1

```
#include <iostream>
using namespace std;

int main ( )
{
    int n, k;

    cout << "Εισάγετε το N ";
    cin >> n;

    k = 1;
    while ( k <= n ) {
        cout << k << endl;
        ++k;
    }
}
```

5

Συντακτικό της εντολής while 1/2

```
while ( συνθήκη )
    εντολή ;
```

Παρατηρήσεις:

- Δεν βάζουμε ερωτηματικό μετά τη σύγκριση.
- Ως *συνθήκη* μπορούμε να χρησιμοποιήσουμε οποιαδήποτε απλή ή σύνθετη λογική παράσταση.
- Συντακτικά η while (μαζί με την εντολή που περιλαμβάνει) θεωρείται ως μία εντολή.
- Ανάλογα με τη *συνθήκη* μπορεί να μην γίνει καμία επανάληψη.

6

Συντακτικό της εντολής while 2/2

```
while ( συνθήκη ) {
    εντολή ;
    εντολή ;
    ⋮
}
```

Σε περίπτωση που θέλουμε να εκτελεστούν **περισσότερες από μια εντολές**, χρησιμοποιούμε άγκιστρα για να ομαδοποιήσουμε τις εντολές.

7

Πως λειτουργεί η εντολή while

```
while ( συνθήκη )
    εντολή ;
```

Αρχικά υπολογίζεται η **συνθήκη**.

Εάν η **συνθήκη** είναι **αληθής** εκτελείται η *εντολή* και το πρόγραμμα επιστρέφει στο while.

Εάν η **συνθήκη** είναι **ψευδής** τερματίζεται η λειτουργία της εντολής while.

8

Παράδειγμα #1 (ξανά)

Μπορούμε να διακρίνουμε τρία σημεία στην εντολή while:

```

k = 1;
while ( k <= n ) {
    cout << k << endl;
    ++k;
}
  
```

Αρχική τιμή
 Συνθήκη
 Αύξηση

9

ΥΠΟΛΟΓΙΣΤΕΣ ΙΙ

Εντολή for

10

Τι χρειάζεται ;

Η εντολή for ενσωματώνει τα τρία σημεία της εντολής while που είδαμε στο παράδειγμα:

- Αρχική τιμή
- Συνθήκη επανάληψης
- Αύξηση της μεταβλητής

11

Παράδειγμα #1 (με for)

```

#include <iostream>
using namespace std;

int main ( )
{
    int n, k;

    cout << "Εισάγετε το N ";
    cin >> n;

    for ( k = 1 ; k <= n ; ++k )
        cout << k << endl;
}
  
```

12

Συντακτικό της εντολής for 1/2

for (αρχική τιμή ; συνθήκη ; αύξηση)
εντολή ;

Παρατηρήσεις:

- Ως **αρχική τιμή** μπορούμε να βάλουμε οποιαδήποτε εντολή ανάθεσης τιμής.
- Ως **συνθήκη** μπορούμε να χρησιμοποιήσουμε οποιαδήποτε απλή ή σύνθετη λογική παράσταση.
- Ως **αύξηση** βάζουμε οποιαδήποτε εντολή που τροποποιεί (αυξάνει ή μειώνει) την τιμή της μεταβλητής.
- Συντακτικά η for (μαζί με την εντολή που περιλαμβάνει) θεωρείται ως μία εντολή.
- Οποιοδήποτε τμήμα από τα **αρχική τιμή, συνθήκη, αύξηση** μπορεί να παραληφθεί.

13

Συντακτικό της εντολής for 2/2

Σε περίπτωση που θέλουμε να εκτελεστούν **περισσότερες από μια εντολές**, χρησιμοποιούμε άγκιστρα για να ομαδοποιήσουμε τις εντολές.

```
for ( αρχική τιμή ; συνθήκη ; αύξηση ) {
    εντολή ;
    εντολή ;
    :
}
```

14

Πως λειτουργεί η εντολή for 1/2

for (αρχική τιμή ; συνθήκη ; αύξηση)
εντολή ;

Αρχικά ανατίθεται η **αρχική τιμή**.

Ελέγχεται η **συνθήκη**.

Εάν η **συνθήκη** είναι **αληθής** εκτελείται η **εντολή**, εκτελείται η **αύξηση** και το πρόγραμμα επιστρέφει στην for για να ελέγξει και πάλι τη **συνθήκη**.

Εάν η **συνθήκη** είναι **ψευδής** τερματίζεται η λειτουργία της εντολής for.

15

Πως λειτουργεί η εντολή for 2/2

Προσοχή:

Στην εντολή for **δεν γράφουμε** την αρχική τιμή, τελική τιμή και βήμα αύξησης της μεταβλητής.

Γράφουμε:

- Μια εντολή για να αναθέσουμε αρχική τιμή.
- Μια συνθήκη που όσο είναι αληθής γίνονται επαναλήψεις.
- Μια εντολή που αυξάνει ή μειώνει την τιμή της μεταβλητής.

16

Σχέση των εντολών for και while

Η παρακάτω εντολή for:

```
for ( αρχική τιμή ; συνθήκη ; αύξηση )
    εντολή ;
```

Είναι ισοδύναμη με:

```
αρχική τιμή ;
while ( συνθήκη ) {
    εντολή ;
    αύξηση ;
}
```

17

Παράδειγμα #2

Η παρακάτω εντολή for:

```
for ( k = 0 ; k < 10 ; ++k )
    cout << k << endl ;
```

Είναι ισοδύναμη με:

```
k = 0;
while ( k < 10 ) {
    cout << k << endl ;
    ++k;
}
```

18

Παράδειγμα #3

Πόσες επαναλήψεις θα γίνουν από τις παρακάτω εντολές for ; Ποια είναι η τιμή της μεταβλητής σε κάθε επανάληψη ;

- 1) for (k=1; k<=7; k=k+2)
- 2) for (k=10; k<17; k+=3)
- 3) for (m=-2; m<2; m++)
- 4) for (m=1; m<=0; m+=25)
- 5) for (n=-8; n>=-10; n=n+2)

19

Παράδειγμα #3

- 1) for (k=1; k<=7; k=k+2)
4 επαναλήψεις: 1, 3, 5, 7
- 2) for (k=10; k<17; k+=3)
3 επαναλήψεις: 10, 13, 16
- 3) for (m=-2; m<2; m++)
4 επαναλήψεις: -2, -1, 0, 1
- 4) for (m=1; m<=0; m+=25)
Καμία επανάληψη
- 5) for (n=-8; n>=-10; n=n+2)
Άπειρες επαναλήψεις

20

Παράδειγμα #4

Πόσες επαναλήψεις θα γίνουν από τις παρακάτω εντολές for ; Ποια είναι η τιμή της μεταβλητής σε κάθε επανάληψη ;

- 1) `for (k=1; k<=15 ;)`
- 2) `for (k=0; ; ++k)`
- 3) `for (; ;)`

21

Παράδειγμα #4

- 1) `for (k=1; k<=15 ;)`
Άπειρες επαναλήψεις
- 2) `for (k=0; ; ++k)`
Άπειρες επαναλήψεις
- 3) `for (; ;)`
Άπειρες επαναλήψεις

22

ΥΠΟΛΟΓΙΣΤΕΣ ΙΙ

Εντολή *do-while*

23

Συντακτικό της εντολής *do-while* 1/2

```
do
    εντολή ;
while ( συνθήκη );
```

Παρατηρήσεις:

- Βάζουμε ερωτηματικό μετά την *εντολή*, αλλά και στο τέλος της *do-while*.
- Συντακτικά η *do-while* (μαζί με την εντολή που περιλαμβάνει) θεωρείται ως μία εντολή.

24

Συντακτικό της εντολής do-while 2/2

Σε περίπτωση που θέλουμε να εκτελεστούν **περισσότερες από μια εντολές**, χρησιμοποιούμε άγκιστρα για να ομαδοποιήσουμε τις εντολές.

```
do {
    εντολή ;
    εντολή ;
    ⋮
} while ( συνθήκη );
```

25

Πως λειτουργεί η εντολή do-while 1/2

```
do
    εντολή ;
while ( συνθήκη );
```

Αρχικά εκτελείται η **εντολή**.

Κατόπιν υπολογίζεται η **συνθήκη**.

Εάν η **συνθήκη** είναι **αληθής** το πρόγραμμα επιστρέφει στο do.

Εάν η **συνθήκη** είναι **ψευδής** τερματίζεται η λειτουργία της εντολής do-while.

26

Πως λειτουργεί η εντολή do-while 2/2

Προσοχή:

Στην εντολή do-while η **εντολή** εκτελείται τουλάχιστον μία φορά όποια και αν είναι η **συνθήκη**.

Στις εντολές while και for η **εντολή** μπορεί να μην εκτελεστεί και καμία φορά ανάλογα με τη **συνθήκη**.

27

Παράδειγμα #5

Μια συνηθισμένη χρήση της do-while:

Σε περίπτωση εισαγωγής λανθασμένων δεδομένων να ζητείται ξανά η εισαγωγή τους.

28

Παράδειγμα #5

Ας υποθέσουμε ότι κάποιο πρόβλημα ζητάει να εισάγουμε ένα θετικό ακέραιο:

```
cout << "Εισάγετε ένα θετικό ακέραιο ";
cin >> n;
```

29

Παράδειγμα #5

Για να ελέγξουμε αν ο χρήστης πληκτρολόγησε ένα θετικό ακέραιο και να ζητήσουμε ξανά τον αριθμό σε περίπτωση λάθους:

```
do {
    cout << "Εισάγετε ένα θετικό ακέραιο ";
    cin >> n;
    if ( n <= 0 ) cout << "ΛΑΘΟΣ \n";
} while ( n <= 0 );
```

Πρακτικός κανόνας:

Στις ανωτέρω εντολές if και do-while γράφω την ίδια συνθήκη.

30

Η εντολή *break*

Αν για οποιοδήποτε λόγο χρειάζεται να "βγούμε" από μια εντολή επανάληψης, χρησιμοποιούμε την εντολή *break*. Συνήθως χρησιμοποιείται με κάποια εντολή *if*.

Παράδειγμα:

```
for ( k=1 ; ; ++k ) {
    :
    if ( ... ) break;
}
```

31

Παράδειγμα #6

Υπολογίστε το άθροισμα

$$\sum_{k=1}^N \frac{1}{k}$$

μόνο για τις περιττές τιμές *k*.

Σημείωση:

Το ζητούμενο άθροισμα είναι:

$$\frac{1}{1} + \frac{1}{3} + \frac{1}{5} + \dots + \frac{1}{N}$$

32

Παράδειγμα #6

```
s = 0 ; // Αθροιστής
for ( k=1 ; k<=n ; k+=2 )
    s += 1.0/k ;
```

33

Παράδειγμα #7

Υπολογίστε το άθροισμα:

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots - \frac{1}{N}$$

Σημείωση:

Το άθροισμα συγκλίνει στο $\pi/4$.

Παρατήρηση:

Κάθε όρος έχει αντίθετο πρόσημο από τον προηγούμενο.

34

Παράδειγμα #7

```
#include <iostream>
using namespace std;

int main ( )
{
    int k, n;
    double s, p;

    cout << "Εισάγετε το n ";
    cin >> n;

    s = 0;
    p = 1.0; // Αρχικό πρόσημο
    for ( k=1; k<=n ; k+=2 ) {
        s += p/k; // Αθροιση
        p = -p; // Αλλαγή προσήμου
    }
    cout << s << endl;
}
```

35

Παράδειγμα #8

Κατασκευάστε πρόγραμμα που θα υπολογίζει το N-οστό όρο της ακολουθίας:

$$x_{k+1} = \frac{3x_k + 4}{2x_k + 3}$$

με πρώτο όρο $x_1=1$

Σημείωση:

Η ακολουθία αυτή συγκλίνει γρήγορα στην τιμή $\sqrt{2}$

36

Παράδειγμα #8

```
#include <iostream>
using namespace std;

int main ( )
{
    int k, n;
    double x;

    cout << "Πόσοι όροι? ";
    cin >> n;

    x = 1.0;
    for ( k=2; k<=n; ++k )
        x = (3*x+4)/(2*x+3);

    cout << x << endl;
}
```

37

Παράδειγμα #9

Ακολουθία Fibonacci

Υπολογίστε το N-στο όρο της ακολουθίας:

$$F_k = F_{k-1} + F_{k-2}$$

με $F_0 = 0$ και $F_1 = 1$

Χρησιμοποιούμε δύο μεταβλητές:

- fb για τον προηγούμενο όρο (F_{k-1})
- fa για τον προ-προηγούμενο όρο (F_{k-2})

38

Παράδειγμα #9

```
#include <iostream>
using namespace std;

int main ( )
{
    int fa, fb, fc;
    int n, i;

    cout << "Ποιόν όρο? ";
    cin >> n;

    fa = 0;
    fb = 1;
    for ( i=2; i<=n; ++i ) {
        fc = fa+fb;    // Ο νέος όρος.
        fa = fb;      // Ο προηγούμενος γίνεται προ-προηγούμενος
        fb = fc;      // Ο νέος όρος γίνεται προηγούμενος.
    }
    cout << fc << endl;
}
```

39

Παράδειγμα #10

Υπολογισμός ολοκληρώματος

Υπολογίστε προσεγγιστικά το ολοκλήρωμα

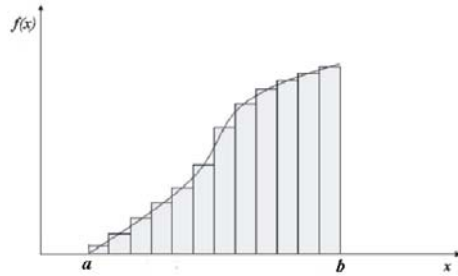
$$I = \int_a^b f(x) dx$$

όταν δίνεται η συνάρτηση $f(x)$ και τα όρια ολοκλήρωσης a, b χρησιμοποιώντας τον **κανόνα του παραλληλογράμμου**.

Το ζητούμενο ολοκλήρωμα ισούται με το εμβαδόν της περιοχής μεταξύ της συνάρτησης και του άξονα x .

40

Παράδειγμα #10



Αλγόριθμος:

1. Χωρίζουμε το διάστημα $[a, b]$ σε N ίσα υποδιαστήματα.
2. Υπολογίζουμε το εμβαδόν του κάθε ορθογωνίου παραλληλογράμμου θεωρώντας ως ύψος την τιμή της συνάρτησης στο μέσον του.
3. Αθροίζουμε όλα τα επιμέρους εμβαδά.

41

Παράδειγμα #10

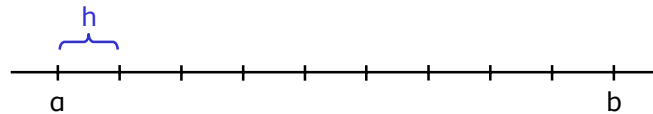
Ποιο είναι το μήκος κάθε υποδιαστήματος ;

$$h = \frac{b-a}{N}$$

42

Παράδειγμα #10

Ποιο είναι το μέσον του k υποδιαστήματος ;



- | | |
|---|------------|
| Το 1 ^ο υποδιάστημα ξεκινάει στο: | a |
| Το 2 ^ο υποδιάστημα ξεκινάει στο: | $a+h$ |
| Το 3 ^ο υποδιάστημα ξεκινάει στο: | $a+2h$ |
| Το 4 ^ο υποδιάστημα ξεκινάει στο: | $a+3h$ |
| ... | |
| Το k υποδιάστημα ξεκινάει στο: | $a+(k-1)h$ |

Το μέσον του k υποδιαστήματος είναι:

$$a+(k-1)h + \frac{h}{2} = a+kh - \frac{h}{2}$$

43

Παράδειγμα #10

Ποιο είναι το εμβαδόν του υποδιαστήματος k ;

$$E_k = h f\left(a+kh - \frac{h}{2}\right)$$

Ποιο είναι το συνολικό εμβαδόν ;

$$\sum_{k=1}^N E_k = \sum_{k=1}^N h f\left(a+kh - \frac{h}{2}\right)$$

44

Παράδειγμα #10

```
#include <iostream>
#include <cmath>
using namespace std;

int main ( )
{
    double a, b;    // Τα όρια ολοκλήρωσης
    double h;      // Μήκος υποδιαστήματος
    double x, f;   // Μέσον υποδιαστήματος και τιμή συνάρτησης
    double s;      // Τιμή ολοκληρώματος
    int n;         // Πλήθος υποδιαστημάτων
    int k;

    cout << "Εισάγετε τα όρια a, b \n";
    cin >> a >> b;
    cout << "Πλήθος διαστημάτων? \n";
    cin >> n;
```

Συνεχίζεται...

45

Παράδειγμα #10

```
...Συνέχεια

h = (b-a)/n;           // Μήκος υποδιαστήματος
s = 0.;
for ( k=1; k<=n; ++k ) {
    x = a + k*h - h/2; // Μέσον του υποδιαστήματος
    f = sin(x);        // Τιμή της συνάρτησης
    s += h*f;          // Άθροιση επιμέρους εμβαδού
}
cout << s << endl;
```

46

Παράδειγμα #11

Υπολογισμός διπλού ολοκληρώματος

Υπολογίστε προσεγγιστικά το διπλό ολοκλήρωμα

$$I = \int_{ya}^{yb} dy \int_{xa}^{xb} f(x, y) dx$$

όταν δίνεται η συνάρτηση $f(x,y)$ και τα όρια ολοκλήρωσης x_a, x_b, y_a, y_b χρησιμοποιώντας τον κανόνα του παραλληλογράμμου.

Εφαρμόζουμε την τεχνική του απλού ολοκληρώματος δύο φορές.

47

Παράδειγμα #11

Μπορούμε να θεωρήσουμε το διπλό ολοκλήρωμα ως:

$$I = \int_{ya}^{yb} dy g(y)$$

όπου η συνάρτηση προς ολοκλήρωση $g(y)$ είναι:

$$g(y) = \int_{xa}^{xb} f(x, y) dx$$



Εφαρμόζουμε την τεχνική του απλού ολοκληρώματος δύο φορές.

48

Παράδειγμα #11

```
#include <iostream>
#include <cmath>
using namespace std;

int main ( )
{
    double xa, xb;           // Τα όρια ολοκλήρωσης στο x
    double ya, yb;           // Τα όρια ολοκλήρωσης στο y
    int n;                   // Πλήθος υποδιαστημάτων
    double xh, yh;           // Μήκη υποδιαστημάτων
    double sx, s;
    int i, j;
    double x, y, f;

    cout << "Εισάγετε τα όρια ";
    cin >> xa >> xb >> ya >> yb ;
    cout << "Πόσα υποδιαστήματα? ";
    cin >> n ;
```

Συνεχίζεται...

49

Παράδειγμα #11

...Συνέχεια

```
xh = (xb-xa)/n;           // Μήκος διαστήματος στο x
yh = (yb-ya)/n;           // Μήκος διαστήματος στο y
s = 0;
for ( j=1; j<=n; ++j ) {
    y = ya+j*yh-yh/2;
    sx = 0;
    for ( i=1; i<=n; ++i ) {
        x = xa+i*xh-xh/2;
        f = x*x*sin(x*y); // Υπολογισμός συνάρτησης
        sx += xh*f;
    }
    s += yh*sx;
}
cout << s << endl;
```

50

Παράδειγμα #12

Υπολογίστε το άθροισμα

$$1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots + \frac{x^N}{N!}$$

όταν δίνεται το x και το N.

Υπενθύμιση:

Το ανωτέρω άθροισμα είναι η σειρά Taylor του e^x

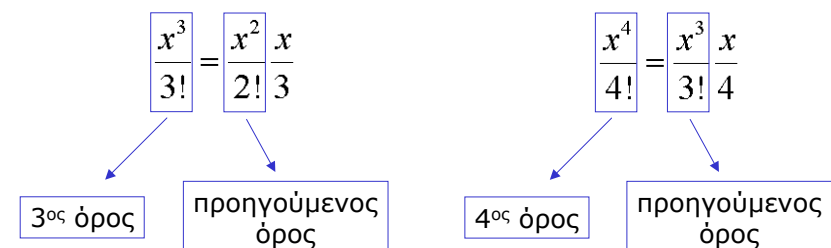
51

Παράδειγμα #12

$$1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots + \frac{x^N}{N!}$$

Παρατηρείστε ότι ο κάθε όρος του αθροίσματος μπορεί να γραφεί σε σχέση με τον προηγούμενο.

Παράδειγμα:



52

Παράδειγμα #12

Γενικά: $\frac{x^k}{k!} = \frac{x^{k-1}}{(k-1)!} \frac{x}{k}$

$$\acute{\alpha}\rho\omicron\varsigma_k = \frac{x}{k} \acute{\alpha}\rho\omicron\varsigma_{k-1} \quad \acute{\alpha}\rho\omicron\varsigma_0 = 1$$



Αναδρομική σχέση με ένα προηγούμενο όρο.

Προσοχή: δεν ζητούνται οι επιμέρους όροι αλλά το άθροισμά τους.

53

Παράδειγμα #12

```
#include <iostream>
using namespace std;

int main ( )
{
    double x, s, t;
    int n, k;

    cout << "Εισάγετε τα x, n \n";
    cin >> x >> n;

    s = 1.;
    t = 1.;
    for ( k=1; k<=n; ++k ) {
        t *= x/k;
        s += t;
    }
    cout << s << endl;
}
```

54

Παράδειγμα #12 (επίλογος)

Πόσοι όροι χρειάζονται για προσεγγίσουμε το e^x με 6 δεκαδικά ψηφία ;

Αντί να προκαθορίσουμε το N θα μπορούσαμε να συγκρίνουμε εντός του for το άθροισμα με την ακριβή τιμή $\exp(x)$ και να σταματήσουμε όταν η διαφορά γίνει μικρότερη από 10^{-6}

Εναλλακτικά, για να μην χρησιμοποιήσουμε τη συνάρτηση $\exp(x)$, μπορούμε να σταματήσουμε όταν ο προστιθέμενος όρος t είναι τόσο μικρός που δεν μπορεί να επηρεάσει τα προηγούμενα ψηφία.

55

Παράδειγμα #13

Υπολογίστε και πάλι το άθροισμα

$$1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots + \frac{x^N}{N!}$$

όταν δίνεται το x. Ή άθροιση να σταματάει όταν ο προστιθέμενος όρος γίνει μικρότερος από 10^{-6} κατ' απόλυτη τιμή.

56

Παράδειγμα #13

```
#include <iostream>
using namespace std;

int main ( )
{
    double x, e, s, t;
    int k;

    cout << "Εισάγετε το x \n";
    cin >> x;

    s = 1;
    t = 1;
    e = 1.e-6;
    for ( k=1; abs(t)>=e; ++k ) {
        t *= x/k;
        s += t;
    }
    cout << s << endl;
}
```

57

Παράδειγμα #14

Υπολογίστε το άθροισμα

$$1 - \frac{1}{2}x + \frac{1 \cdot 3}{2 \cdot 4}x^2 - \frac{1 \cdot 3 \cdot 5}{2 \cdot 4 \cdot 6}x^3 + \dots$$

μέχρι τον όρο τάξης N, όταν δίνεται το x και το N.

Υπενθύμιση:

Το ανωτέρω άθροισμα αποτελεί μια προσέγγιση του

$$\frac{1}{\sqrt{1+x}} \quad \text{για } -1 < x \leq 1 \quad (\text{διωνυμική σειρά})$$

58

Παράδειγμα #14

$$1 - \frac{1}{2}x + \frac{1 \cdot 3}{2 \cdot 4}x^2 - \frac{1 \cdot 3 \cdot 5}{2 \cdot 4 \cdot 6}x^3 + \dots$$

Παρατηρούμε ότι ο κάθε όρος του αθροίσματος μπορεί να γραφεί σε σχέση με τον προηγούμενο.

Παράδειγμα:

$$\boxed{-\frac{1 \cdot 3 \cdot 5}{2 \cdot 4 \cdot 6}x^3} = \boxed{\frac{1 \cdot 3}{2 \cdot 4}x^2} \left(-\frac{5}{6}x \right)$$

↓
↓

3ος όρος

προηγούμενος όρος

59

Παράδειγμα #14

$$\text{Γενικά: } \frac{1 \cdot 3 \cdot 5 \dots (2k-1)}{2 \cdot 4 \cdot 6 \dots (2k)}x^k = \frac{1 \cdot 3 \cdot 5 \dots 2(k-1)-1}{2 \cdot 4 \cdot 6 \dots 2(k-1)}x^{k-1} \left[-\frac{(2k-1)}{(2k)}x \right]$$

$$\acute{\alpha}\rho\omicron\varsigma_k = \acute{\alpha}\rho\omicron\varsigma_{k-1} \left[-x \frac{2k-1}{2k} \right]$$

$$\acute{\alpha}\rho\omicron\varsigma_0 = 1$$

60

Παράδειγμα #14

```
#include <iostream>
using namespace std;

int main ( )
{
    int k, n;
    double x, s, t;

    cout << "Εισάγετε τα x, n ";
    cin >> x >> n;

    s = 1;
    t = 1;
    for ( k=1; k<=n; ++k ) {
        t = -t*x*(2*k-1)/(2*k);
        s += t;
    }
    cout << s << endl;
}
```