

# ΕΙΣΑΓΩΓΗ ΣΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ (2022-23)

## Εργασία 1

Το αντικείμενο της εργασίας αυτής σχετίζεται με τους πρώτους αριθμούς, την παραγοντοποίηση ακεραίων, δηλαδή την ανάλυσή τους σε γινόμενο πρώτων παραγόντων, καθώς και με την έννοια της διαιρετότητας στους ακεραίους. Πριν παρουσιαστεί αναλυτικά το ζητούμενο της εργασίας, θα δοθεί το απαραίτητο μαθηματικό υπόβαθρο.

### Διαιρέτες ακεραίων αριθμών

Ένας θετικός ακεραίος  $d$  είναι *διαιρέτης* (divisor) ενός θετικού ακεραίου  $N$  όταν το υπόλοιπο της ακεραίας διαίρεσης του  $N$  με το  $d$  ισούται με 0. Κάθε αριθμός έχει ως διαιρέτες του τουλάχιστον το 1 και τον εαυτό του. Για παράδειγμα, οι διαιρέτες του 72 είναι οι 1, 2, 3, 4, 6, 8, 9, 12, 18, 24, 36 και 72.

### Πρώτοι αριθμοί

Ένας ακεραίος αριθμός *μεγαλύτερος του 1* ονομάζεται *πρώτος (prime)* όταν έχει σαν μόνους διαιρέτες το 1 και τον εαυτό του. Για παράδειγμα, το 13 είναι πρώτος αριθμός, ενώ το 15 δεν είναι, αφού έχει σαν διαιρέτες τους 3 και 5, εκτός από τους 1 και 15. Μπορείτε να επιβεβαιώσετε ότι οι πρώτοι αριθμοί που είναι μικρότεροι από το 100 είναι οι εξής:

2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97

### Παραγοντοποίηση σε πρώτους παράγοντες

Παραγοντοποίηση ενός ακεραίου αριθμού *μεγαλύτερου του 1* είναι η ανάλυσή του σε *γινόμενο πρώτων παραγόντων*, δηλαδή η γραφή του στη μορφή  $p_1^{e_1} \cdot p_2^{e_2} \cdot \dots \cdot p_k^{e_k}$ , όπου τα  $p_i$  είναι πρώτοι αριθμοί και τα  $e_i$  είναι ακέραιοι  $\geq 1$ , για  $1 \leq i \leq k$ . Μερικά παραδείγματα παραγοντοποίησης είναι τα εξής:

6	$= 2 \cdot 3$
60	$= 2^2 \cdot 3 \cdot 5$
72	$= 2^3 \cdot 3^2$
187	$= 11 \cdot 17$
5234	$= 2 \cdot 2617$
23157	$= 3^2 \cdot 31 \cdot 83$
128377	$= 128377$
5489126	$= 2 \cdot 2744563$
89000362	$= 2 \cdot 11 \cdot 29 \cdot 199 \cdot 701$
127870872	$= 2^3 \cdot 3 \cdot 17 \cdot 313409$
2001234568	$= 2^3 \cdot 1439 \cdot 173839$

### Αριθμοί ελεύθεροι-τετραγώνου

Ένας θετικός ακεραίος αριθμός ονομάζεται *ελεύθερος-τετραγώνου* (square-free), ή ET για τη συνήθεια, όταν *δεν έχει κανένα διαιρέτη που να είναι τέλειο τετράγωνο*, εκτός, φυσικά, από το 1. Για παράδειγμα το 30 είναι ET αριθμός, ενώ το 18 δεν είναι (αφού έχει σαν διαιρέτη το 9 που είναι τέλειο τετράγωνο). Εύκολα μπορούμε να δούμε ότι ένας αριθμός είναι ET όταν στην ανάλυσή του σε γινόμενο πρώτων παραγόντων δεν υπάρχει κάποιος πρώτος παράγοντας υψωμένος σε δύναμη μεγαλύτερη του 1. Στα παραδείγματα παραγοντοποίησης της προηγούμενης παραγράφου, οι αριθμοί 6, 187, 5234,

128377, 5489126 και 89000362 είναι ET. Προφανώς, κάθε πρώτος αριθμός είναι ET. Οι ET αριθμοί που είναι μικρότεροι από το 40 είναι οι εξής:

1, 2, 3, 5, 6, 7, 10, 11, 13, 14, 15, 17, 19, 21, 22, 23, 26, 29, 30, 31, 33, 34, 35, 37, 38, 39

## Τέλειοι, ελαττωματικοί και άφθονοι αριθμοί

Ορίζουμε τη συνάρτηση  $s(\cdot)$  η οποία όταν εφαρμοσθεί επάνω σε ένα θετικό ακέραιο, επιστρέφει το άθροισμα των διαιρετών του, εξαιρουμένου του εαυτού του. Για παράδειγμα  $s(18) = 1 + 2 + 3 + 6 + 9 = 21$ .

Ένας αριθμός  $n$  ονομάζεται τέλειος (perfect) αν ισχύει  $s(n) = n$ . Για παράδειγμα, ο αριθμός 28 είναι τέλειος, αφού οι διαιρέτες του 1, 2, 4, 7 και 14 έχουν άθροισμα 28.

Ένας αριθμός  $n$  ονομάζεται ελαττωματικός (deficient) αν ισχύει  $s(n) < n$ . Για παράδειγμα, ο αριθμός 99 είναι ελαττωματικός, αφού οι διαιρέτες του 1, 3, 9, 11 και 33 έχουν άθροισμα 57, μικρότερο του 99.

Ένας αριθμός  $n$  ονομάζεται άφθονος (abundant) αν ισχύει  $s(n) > n$ . Για παράδειγμα, ο αριθμός 96 είναι άφθονος, αφού οι διαιρέτες του 1, 2, 3, 4, 6, 8, 12, 16, 24, 32 και 48 έχουν άθροισμα 156, μεγαλύτερο του 96.

## Συνάρτηση Möbius

Η συνάρτηση Möbius  $\mu(\cdot)$  ορίζεται επάνω στο σύνολο των θετικών ακεραίων ως εξής:

$$\mu(n) = \begin{cases} 1, & \text{αν το } n \text{ είναι ET και έχει άρτιο πλήθος πρώτων παραγόντων} \\ -1, & \text{αν το } n \text{ είναι ET και έχει περιττό πλήθος πρώτων παραγόντων} \\ 0, & \text{αν το } n \text{ δεν είναι ET} \end{cases}$$



Κάποια παραδείγματα τιμών της συνάρτησης Möbius είναι:

$\mu(1) = 1$ , διότι το 1 είναι ET και έχει 0 (άρτιος) πρώτους παράγοντες.

$\mu(4) = 0$ , διότι το 4 δεν είναι ET.

$\mu(5) = -1$ , διότι το 5 είναι ET και έχει 1 (περιττός) πρώτο παράγοντα, το 5.

$\mu(6) = 1$ , διότι το 6 είναι ET και έχει 2 (άρτιος) πρώτους παράγοντες, τους 2 και 3.

$\mu(24) = 0$ , διότι το 24 δεν είναι ET.

$\mu(30) = -1$ , διότι το 30 είναι ET και έχει 3 (περιττός) πρώτους παράγοντες, τους 2, 3 και 5.

## Συνάρτηση Mertens

Η συνάρτηση Mertens  $M(\cdot)$  ορίζεται επάνω στο σύνολο των θετικών ακεραίων, μέσω της συνάρτησης Möbius, ως εξής:

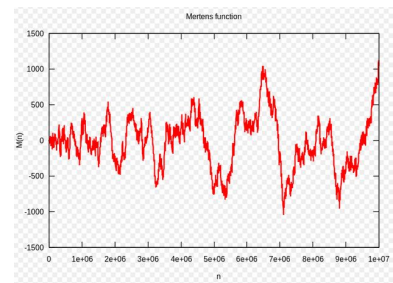
$$M(n) = \sum_{k=1}^n \mu(k)$$

Ουσιαστικά, η συνάρτηση Mertens επιστρέφει, για ένα θετικό ακέραιο  $n$ , τη διαφορά του πλήθους των ET αριθμών που είναι μικρότεροι από ή ίσοι με το  $n$  και έχουν άρτιο πλήθος πρώτων παραγόντων μείον το πλήθος των ET αριθμών που είναι μικρότεροι από ή ίσοι με το  $n$  και έχουν περιττό πλήθος πρώτων παραγόντων.

Οι πρώτες 30 τιμές της συνάρτησης Mertens οι εξής:

1, 0, -1, -1, -2, -1, -2, -2, -2, -1, -2, -2, -3, -2, -1, -1, -2, -2, -3, -3, -2, -1, -2, -2, -2, -1, -1, -1, -2, -3


Η συνάρτηση Mertens έχει πολύ μεγάλο θεωρητικό ενδιαφέρον. Σχετίζεται στενά με την υπόθεση/εικασία του Riemann, που είναι ένα από τα μεγάλα, ίσως το σημαντικότερο, άλυτα προβλήματα των Μαθηματικών. Αυξάνει, κατ' απόλυτη τιμή, αργά προς θετικές και αρνητικές τιμές, παρουσιάζει τοπικά μέγιστα και ελάχιστα και ταλαντώνεται με ένα περίεργο, ίσως και χαοτικό, τρόπο. Στο διάγραμμα δεξιά (από το [https://en.wikipedia.org/wiki/Mertens\\_function](https://en.wikipedia.org/wiki/Mertens_function)), φαίνεται η γραφική παράσταση της συνάρτησης μέχρι το  $n = 10^7$ . Ορίζουμε ως **μηδενικό σημείο** (zero point) της συνάρτησης Mertens κάθε  $n$  για το οποίο ισχύει  $M(n) = 0$ . Τα πρώτα 20 μηδενικά σημεία της συνάρτησης είναι τα εξής:



2, 39, 40, 58, 65, 93, 101, 145, 149, 150, 159, 160, 163, 164, 166, 214, 231, 232, 235, 236

## Επιτέλους, το ζητούμενο της εργασίας

Γράψτε ένα πρόγραμμα C (έστω ότι το πηγαίο αρχείο του ονομάζεται “mertsumd.c”) το οποίο, δεδομένου ενός ακεραίου MAXEXP (συμβολική σταθερά που μπορεί να οριστεί στο πρόγραμμά σας με #define) στο διάστημα  $[1, 9]$ , να εκτελεί τις εξής λειτουργίες.

- Να βρίσκει τις τιμές της συνάρτησης Mertens για κάθε  $n$  στο διάστημα  $[1, 10^{\text{MAXEXP}+9}]$  και να εκτυπώνει εκείνες που αντιστοιχούν στις τιμές του  $n$  που βρίσκονται μέσα στα διαστήματα  $[10^K-9, 10^K+9]$ , για κάθε  $K$  στο διάστημα  $[1, \text{MAXEXP}]$ . 
- Να βρίσκει και να εκτυπώνει το πλήθος, έστω ZP, των **μηδενικών σημείων** της συνάρτησης Mertens στο διάστημα  $[1, 10^{\text{MAXEXP}+9}]$ .
- Να βρίσκει και να εκτυπώνει τους **τέλειους** αριθμούς στο διάστημα  $[2, 1000 \cdot \text{ZP}]$ .
- Να βρίσκει και να εκτυπώνει το πλήθος των **ελαττωματικών** και το πλήθος των **άφθονων αριθμών** στο διάστημα  $[2, 1000 \cdot \text{ZP}]$ .

Για να γίνει απολύτως κατανοητό το ζητούμενο της εργασίας, δίνονται στη συνέχεια τα αποτελέσματα (πλήρως ή μερικώς) από εκτελέσεις ενός ενδεικτικού προγράμματος, για διάφορες τιμές του MAXEXP.

Για να μπορείτε να πειραματίζεστε εύκολα με διαφορετικές τιμές του MAXEXP, χωρίς να χρειάζεται να τροποποιείτε το πηγαίο αρχείο του προγράμματός σας, μπορείτε να μην ορίσετε αυτή τη συμβολική σταθερά μέσα στο αρχείο, αλλά να δίνετε την επιθυμητή τιμή στην εντολή μεταγλώττισης, με τον τρόπο που φαίνεται στη συνέχεια.<sup>1</sup>

```
$ hostname
linux14
$
$ gcc -DMAXEXP=1 -o mertsumd1 mertsumd.c
$ ./mertsumd1
M(1) = 1
M(2) = 0
M(3) = -1
M(4) = -1
M(5) = -2
```

<sup>1</sup>Η εντολή time χρονομετρεί την εντολή που ακολουθεί. Στο πρώτο παράδειγμα από τα επόμενα που χρησιμοποιείται αυτή, το πρόγραμμα χρειάστηκε 3.986 CPU δευτερόλεπτα κατά την εκτέλεσή του. Όλες οι εκτελέσεις έγιναν σε μηχάνημα Linux του εργαστηρίου του Τμήματος.

M(6) = -1  
M(7) = -2  
M(8) = -2  
M(9) = -2  
M(10) = -1  
M(11) = -2  
M(12) = -2  
M(13) = -3  
M(14) = -2  
M(15) = -1  
M(16) = -1  
M(17) = -2  
M(18) = -2  
M(19) = -3

Found 1 zero points of the Mertens function

Checking numbers in the range [2,1000]

Found perfect number: 6  
Found perfect number: 28  
Found perfect number: 496

Found 750 deficient numbers  
Found 246 abundant numbers

\$  
\$ gcc -DMAXEXP=2 -o mertsumd2 mertsumd.c  
\$ ./mertsumd2

M(1) = 1  
M(2) = 0  
M(3) = -1  
M(4) = -1  
M(5) = -2  
M(6) = -1  
M(7) = -2  
M(8) = -2  
M(9) = -2  
M(10) = -1  
M(11) = -2  
M(12) = -2  
M(13) = -3  
M(14) = -2  
M(15) = -1  
M(16) = -1  
M(17) = -2  
M(18) = -2  
M(19) = -3  
.....  
M(91) = -1

M(92) = -1  
M(93) = 0  
M(94) = 1  
M(95) = 2  
M(96) = 2  
M(97) = 1  
M(98) = 1  
M(99) = 1  
M(100) = 1  
M(101) = 0  
M(102) = -1  
M(103) = -2  
M(104) = -2  
M(105) = -3  
M(106) = -2  
M(107) = -3  
M(108) = -3  
M(109) = -4

Found 7 zero points of the Mertens function

Checking numbers in the range [2,7000]

Found perfect number: 6  
Found perfect number: 28  
Found perfect number: 496

Found 5253 deficient numbers  
Found 1743 abundant numbers

```
$  
$ gcc -DMAXEXP=3 -o mertsumd3 mertsumd.c  
$ ./mertsumd3  
M(1) = 1  
M(2) = 0  
M(3) = -1  
M(4) = -1  
M(5) = -2  
M(6) = -1  
M(7) = -2  
M(8) = -2  
M(9) = -2  
M(10) = -1  
M(11) = -2  
M(12) = -2  
M(13) = -3  
M(14) = -2  
M(15) = -1  
M(16) = -1  
M(17) = -2
```

```

M(18) = -2
M(19) = -3
.....
M(91) = -1
M(92) = -1
M(93) = 0
M(94) = 1
M(95) = 2
M(96) = 2
M(97) = 1
M(98) = 1
M(99) = 1
M(100) = 1
M(101) = 0
M(102) = -1
M(103) = -2
M(104) = -2
M(105) = -3
M(106) = -2
M(107) = -3
M(108) = -3
M(109) = -4
.....
M(991) = 1
M(992) = 1
M(993) = 2
M(994) = 1
M(995) = 2
M(996) = 2
M(997) = 1
M(998) = 2
M(999) = 2
M(1000) = 2
M(1001) = 1
M(1002) = 0
M(1003) = 1
M(1004) = 1
M(1005) = 0
M(1006) = 1
M(1007) = 2
M(1008) = 2
M(1009) = 1

```

Found 94 zero points of the Mertens function

Checking numbers in the range [2,94000]

Found perfect number: 6

Found perfect number: 28

Found perfect number: 496  
Found perfect number: 8128

Found 70689 deficient numbers  
Found 23306 abundant numbers

```
$  
$ gcc -DMAXEXP=4 -o mertsumd4 mertsumd.c  
$ ./mertsumd4  
..... (omitted to save space)  
M(10006) = -20  
M(10007) = -21  
M(10008) = -21  
M(10009) = -22
```

Found 406 zero points of the Mertens function

Checking numbers in the range [2,406000]

Found perfect number: 6  
Found perfect number: 28  
Found perfect number: 496  
Found perfect number: 8128

Found 305521 deficient numbers  
Found 100474 abundant numbers

```
$  
$ gcc -DMAXEXP=5 -o mertsumd5 mertsumd.c  
$ ./mertsumd5  
..... (omitted to save space)  
M(100006) = -47  
M(100007) = -46  
M(100008) = -46  
M(100009) = -46
```

Found 1549 zero points of the Mertens function

Checking numbers in the range [2,1549000]

Found perfect number: 6  
Found perfect number: 28  
Found perfect number: 496  
Found perfect number: 8128

Found 1165576 deficient numbers  
Found 383419 abundant numbers

```
$  
$ gcc -DMAXEXP=6 -o mertsumd6 mertsumd.c  
$ time ./mertsumd6  
..... (omitted to save space)
```

```
M(1000006) = 211
M(1000007) = 212
M(1000008) = 212
M(1000009) = 213
```

Found 5361 zero points of the Mertens function

Checking numbers in the range [2,5361000]

```
Found perfect number: 6
Found perfect number: 28
Found perfect number: 496
Found perfect number: 8128
```

Found 4033557 deficient numbers

Found 1327438 abundant numbers

```
3.986u 0.000s 0:03.98 100.0%    0+0k 0+0io 0pf+0w
```

```
$
```

```
$ gcc -DMAXEXP=7 -o mertsumd7 mertsumd.c
```

```
$ time ./mertsumd7
```

```
..... (omitted to save space)
```

```
M(10000006) = 1040
M(10000007) = 1041
M(10000008) = 1041
M(10000009) = 1042
```

Found 12546 zero points of the Mertens function

Checking numbers in the range [2,12546000]

```
Found perfect number: 6
Found perfect number: 28
Found perfect number: 496
Found perfect number: 8128
```

Found 9438518 deficient numbers

Found 3107477 abundant numbers

```
17.060u 0.000s 0:17.06 100.0%    0+0k 0+0io 0pf+0w
```

```
$
```

```
$ gcc -DMAXEXP=8 -o mertsumd8 mertsumd.c
```

```
$ ./mertsumd8
```

```
..... (omitted to save space)
```

```
M(100000006) = 1929
M(100000007) = 1928
M(100000008) = 1928
M(100000009) = 1929
```

Found 41908 zero points of the Mertens function



Checking numbers in the range [2,41908000]

Found perfect number: 6

Found perfect number: 28

Found perfect number: 496

Found perfect number: 8128

Found perfect number: 33550336

Found 31528433 deficient numbers

Found 10379561 abundant numbers

\$

\$ gcc -DMAXEXP=9 -o mertsumd9 mertsumd.c

\$ ./mertsumd9

M(1) = 1

M(2) = 0

M(3) = -1

M(4) = -1

M(5) = -2

M(6) = -1

M(7) = -2

M(8) = -2

M(9) = -2

M(10) = -1

M(11) = -2

M(12) = -2

M(13) = -3

M(14) = -2

M(15) = -1

M(16) = -1

M(17) = -2

M(18) = -2

M(19) = -3

.....

M(91) = -1

M(92) = -1

M(93) = 0

M(94) = 1

M(95) = 2

M(96) = 2

M(97) = 1

M(98) = 1

M(99) = 1

M(100) = 1

M(101) = 0

M(102) = -1

M(103) = -2

M(104) = -2

M(105) = -3

M(106) = -2

M(107) = -3  
M(108) = -3  
M(109) = -4

.....

M(991) = 1  
M(992) = 1  
M(993) = 2  
M(994) = 1  
M(995) = 2  
M(996) = 2  
M(997) = 1  
M(998) = 2  
M(999) = 2  
M(1000) = 2  
M(1001) = 1  
M(1002) = 0  
M(1003) = 1  
M(1004) = 1  
M(1005) = 0  
M(1006) = 1  
M(1007) = 2  
M(1008) = 2  
M(1009) = 1

.....

M(9991) = -26  
M(9992) = -26  
M(9993) = -25  
M(9994) = -26  
M(9995) = -25  
M(9996) = -25  
M(9997) = -24  
M(9998) = -23  
M(9999) = -23  
M(10000) = -23  
M(10001) = -22  
M(10002) = -23  
M(10003) = -22  
M(10004) = -22  
M(10005) = -21  
M(10006) = -20  
M(10007) = -21  
M(10008) = -21  
M(10009) = -22

.....

M(99991) = -49  
M(99992) = -49  
M(99993) = -48  
M(99994) = -48  
M(99995) = -49

M(99996) = -49  
M(99997) = -49  
M(99998) = -48  
M(99999) = -48  
M(100000) = -48  
M(100001) = -47  
M(100002) = -46  
M(100003) = -47  
M(100004) = -47  
M(100005) = -46  
M(100006) = -47  
M(100007) = -46  
M(100008) = -46  
M(100009) = -46  
.....  
M(999991) = 210  
M(999992) = 210  
M(999993) = 211  
M(999994) = 210  
M(999995) = 211  
M(999996) = 211  
M(999997) = 212  
M(999998) = 212  
M(999999) = 212  
M(1000000) = 212  
M(1000001) = 213  
M(1000002) = 212  
M(1000003) = 211  
M(1000004) = 211  
M(1000005) = 212  
M(1000006) = 211  
M(1000007) = 212  
M(1000008) = 212  
M(1000009) = 213  
.....  
M(9999991) = 1034  
M(9999992) = 1034  
M(9999993) = 1035  
M(9999994) = 1034  
M(9999995) = 1035  
M(9999996) = 1035  
M(9999997) = 1036  
M(9999998) = 1037  
M(9999999) = 1037  
M(10000000) = 1037  
M(10000001) = 1038  
M(10000002) = 1039  
M(10000003) = 1040  
M(10000004) = 1040

```

M(100000005) = 1039
M(100000006) = 1040
M(100000007) = 1041
M(100000008) = 1041
M(100000009) = 1042
.....
M(999999991) = 1925
M(999999992) = 1925
M(999999993) = 1926
M(999999994) = 1925
M(999999995) = 1926
M(999999996) = 1926
M(999999997) = 1927
M(999999998) = 1928
M(999999999) = 1928
M(1000000000) = 1928
M(1000000001) = 1929
M(1000000002) = 1928
M(1000000003) = 1929
M(1000000004) = 1929
M(1000000005) = 1930
M(1000000006) = 1929
M(1000000007) = 1928
M(1000000008) = 1928
M(1000000009) = 1929
.....
M(9999999991) = -219
M(9999999992) = -219
M(9999999993) = -220
M(9999999994) = -221
M(9999999995) = -220
M(9999999996) = -220
M(9999999997) = -221
M(9999999998) = -222
M(9999999999) = -222
M(10000000000) = -222
M(10000000001) = -223
M(10000000002) = -224
M(10000000003) = -225
M(10000000004) = -225
M(10000000005) = -226
M(10000000006) = -225
M(10000000007) = -226
M(10000000008) = -226
M(10000000009) = -227

```

Found 141121 zero points of the Mertens function

Checking numbers in the range [2,141121000]

Found perfect number: 6  
Found perfect number: 28  
Found perfect number: 496  
Found perfect number: 8128  
Found perfect number: 33550336

Found 106179364 deficient numbers  
Found 34941630 abundant numbers

### Σημειώσεις/Υποδείξεις/Απαγορεύσεις

- Προτείνεται να αναπτύξετε το πρόγραμμά σας σταδιακά. Δηλαδή, να υλοποιείτε τις ζητούμενες λειτουργίες μία-μία, όπως έχουν περιγραφεί προηγουμένως, και, αφού εξασφαλίζετε ότι τα αποτελέσματά σας είναι σωστά, να συνεχίζετε με την επόμενη. Επίσης, αρχικά να δοκιμάζετε μικρές τιμές του MAXEXP, ώστε να μπορείτε εύκολα να βρίσκετε και να διορθώνετε ενδεχόμενα λάθη στο πρόγραμμά σας.
- Στο πρόγραμμα που θα παραδώσετε, να θέσετε ως τιμή στη συμβολική σταθερά MAXEXP τη μέγιστη για την οποία το πρόγραμμα να τερματίζει σε λιγότερο από 200 secs σε μηχάνημα Linux του εργαστηρίου του Τμήματος (χωρίς χρήση επιλογών βελτιστοποίησης κατά τη μεταγλώττιση από τον gcc).
- Για τη διευκόλυνση του ελέγχου ορθότητας των αποτελεσμάτων του προγράμματος που θα παραδώσετε, πρέπει οι εκτυπώσεις να είναι **ακριβώς** στη μορφή που φαίνεται στις εκτελέσεις που δόθηκαν προηγουμένως (εννοείται ότι στη θέση των γραμμών που έχουν τη σημείωση “omitted to save space” πρέπει να δίνονται τα πραγματικά αποτελέσματα του προγράμματος στην επιθυμητή μορφή).
- Επισημαίνεται ότι οι εκτυπώσεις από τις εκτελέσεις του ενδεικτικού προγράμματος δίνονται απλώς και μόνο για να μπορείτε να ελέγξετε την ορθότητα των αποτελεσμάτων του δικού σας προγράμματος. **Δεν επιτρέπεται να χρησιμοποιήσετε στο πρόγραμμά σας ως δεδομένο οτιδήποτε υπάρχει στις εκτυπώσεις που έχουν δοθεί.**
- Στην εργασία αυτή **απαγορεύονται αυστηρά η χρήση αριθμών και μεταβλητών κινητής υποδιαστολής, πινάκων, δεικτών και συναρτήσεων της μαθηματικής βιβλιοθήκης της C.** Επίσης, **δεν επιτρέπεται ο ορισμός άλλων συναρτήσεων πλην της main().**
- Η παράδοση της εργασίας αυτής συνίσταται στην υποβολή του πηγαίου αρχείου mertsumd.c μέσω του e-class του μαθήματος (επιλογή “Εργασίες”).