ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ

Τμήμα Εφαρμοσμένης Πληροφορικής

ΔΙΑΔΙΚΑΣΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Εξάμηνο Α'

Φύλλο Ασκήσεων 4 – ΣΥΝΑΡΤΗΣΕΙΣ - ΔΙΑΔΙΚΑΣΙΕΣ

Διδάσκοντες: Μάγια Σατρατζέμη, Αλέξανδρος Χατζηγεωργίου, Στέλιος Ξυνόγαλος, Ηλίας Σακελλαρίου, Αλέξανδρος Καρακασίδης

Παρατηρήσεις:

1. Τα δεδομένα εισόδου διαβάζονται με τη σειρά που δηλώνονται στις εκφωνήσεις. Για κάθε δεδομένο εισόδου να χρησιμοποιείτε προτρεπτικό μήνυμα που θα ενημερώνει τον χρήστη για την τιμή που αναμένεται.
2. Αντίστοιχα για τα δεδομένα εξόδου και όπου δεν υπάρχουν περαιτέρω διευκρινήσεις για τη μορφή τους, αυτά θα εμφανίζονται με ξεχωριστές εντολές `printf("…\n")` το καθένα και με τη σειρά που δηλώνονται στις εκφωνήσεις.
3. Τα αριθμητικά δεδομένα αναπαρίστανται πάντα από μεταβλητές ακέραιου τύπου (`int` ή `long`). Σε αντίθετη περίπτωση (μεταβλητές τύπου `double/float`) θα γίνονται οι απαραίτητες διευκρινήσεις.
4. Για την εμφάνιση των τιμών μεταβλητών τύπου `double/float` τα δεδομένα θα εμφανίζονται με ένα δεκαδικό ψηφίο. Σε περίπτωση που απαιτείται διαφορετική στοίχιση ή διαφορετική ακρίβεια θα δίνονται οι απαραίτητες διευκρινήσεις.
5. Για την εμφάνιση πολλών δεδομένων στην ίδια γραμμή θα τυπώνεται ένας κενός χαρακτήρας ανά δεδομένο. Για την αναπαράσταση του κενού χαρακτήρα στις εκφωνήσεις χρησιμοποιείται η κάτω παύλα -underscore- "_".

---

1. Δίνεται ο παρακάτω σκελετός ενός προγράμματος C:

```
const
   pi = 3.14159,
   two = 2;

   int Month, Day, Year, p, q;
   doubleHours, Rate, Amount, u, w;
   char Code, Class;

void Calculate (double a, double B, int m,  int k,  int n, int c);
                        . . . . . .
```

Προσδιορίστε ποιες από τις επόμενες εντολές μπορούν να χρησιμοποιηθούν μέσα σε ένα πρόγραμμα (δηλαδή δεν περιέχουν λάθη):

a. `Calculate(u, v, two, p, q, Code);`
b. `Calculate(pi, u, w, two, Day, Year, Class);`
c. `Calculate(Hours, pi, two, Day, Year, Class);`
d. `While (u>0) Calculate(Rate, u, Day+2, p, q, 'm');`
e. `Calculate(0, Hours, (p+1)/2, Day, year, Code);`
f. `while (Amount>0) Calculate(two, Amount, Day, p+q, Day, Class);`

2. Δίνεται ο παρακάτω σκελετός ενός προγράμματος C:

```
main()
{
     const  pi = 3.14159,  two = 2;

          int Month, Day, Year, p, q;
               double Hours, Rate, Amount, u, v;
               char Code, Class;
               . . . . . .
}
double f(double x, double y, int d)
{
  . . . . . .
```

}

Προσδιορίστε ποιες από τις επόμενες εντολές μπορούν να χρησιμοποιηθούν μέσα σε αυτό το πρόγραμμα (δηλαδή δεν περιέχουν συντακτικά λάθη):

a. `Amount = f(pi, Rate, month);`

b. `Rate = f(Hours, Day, two);`

c. `printf("%lf", f(0, 0, 0));`

d. `f(Hours, Rate, Month);`

e. `Hours = two*f(pi, amount) / (2.71828*Rate);`

f. `Amount = f(pi*Hours, (2.71828+Day) / Rate, two);`

g.
```
if (Month== two)
    a. Year = f(Hours, f(Rate, pi, two), Day);
```

h.
```
k) while (f(Amount, 0, 0) < pi)
    a. Amount = f(Amount, pi, 1);
```

i.
```
l) do
    a. Amount = f(Amount, 0, code)
```

j. `while (Amount > 0);`

k. `n) Amount = f(a, b, day);`

3. Να γραφεί ένα πρόγραμμα που θα διαβάζει έναν αριθμό τύπου `double` (μεταβλητή `number`) και έναν αριθμό τύπου `int` (μεταβλητή `places`) και θα περιλαμβάνει μια συνάρτηση με όνομα `ceil_dec(double number, int places)`, η οποία θα επιστρέφει την τιμή της μεταβλητής `number` στρογγυλοποιημένη με τον τρόπο που καθορίζει η συνάρτησηση `ceil()` στον καθορισμένο αριθμό δεκαδικών ψηφίων `places` (π.χ. `ceil_dec(12.547,0)`, `ceil_dec(12.547,1)`, `ceil_dec (12.547,2)` θα επιστρέφουν 13, 12.6 και 12.55 αντίστοιχα). Η εμφάνιση της στρογγυλοποιημένης τιμής της μεταβλητής `number` θα γίνεται από το κυρίως πρόγραμμα με στοίχιση αριστερά χωρίς κενά. *Υπόδειξη: Η συνάρτηση `ceil_dec` θα κάνει χρήση των συναρτήσεων `pow` και `ceil` της `math.h`.*

4. Να γραφεί πρόγραμμα που θα διαβάζει τις τιμές τριών μεταβλητών `Kefalaio` (τύπου `long`), `Epitokio` (τύπου `double`, π.χ. για επιτόκιο 12% ⇨ 0.12) και `Eti` (τύπου `int`) και στη συνέχεια θα υπολογίζει και θα εμφανίζει το συνολικό κεφάλαιο (τύπου `long`). Η εισαγωγή των στοιχείων θα γίνεται στο κυρίως πρόγραμμα και ο υπολογισμός του κεφαλαίου θα γίνεται με τη βοήθεια συνάρτησης. Για τον υπολογισμό του ετήσιου κεφαλαίου θα χρησιμοποιείται ο τύπος `ceil(Κεφάλαιο*(1+Επιτόκιο))`.

```
Dwse to kefalaio: 100000
Dwse to eptokio: 0.1
Dwse ta eth: 3
133103
```

5. Οι αριθμοί για τους οποίους το άθροισμα των κύβων των ψηφίων τους είναι ίσο με τον ίδιο τον αριθμό λέγονται αριθμοί Armstrong (π.χ. $153 = 1^3 + 5^3 + 3^3$). Να γραφεί πρόγραμμα που θα περιλαμβάνει τα παρακάτω:

- την συνάρτηση `Sum_Cube` που θα δέχεται έναν ακέραιο αριθμό (μεταξύ του 1 και του 999) και θα επιστρέφει το άθροισμα των κύβων των ψηφίων του.
- την συνάρτηση `IsArmstrong` που θα δέχεται έναν ακέραιο αριθμό και θα επιστρέφει την τιμή `TRUE` ή `FALSE` ανάλογα με το αν ο αριθμός είναι Armstrong ή όχι αντίστοιχα.

Το κυρίως πρόγραμμα θα βρίσκει και θα εμφανίζει ποιοι είναι αριθμοί Armstrong στο διάστημα (1-999).

```
1
153
370
371
407
```

6. Να γραφεί ένα πρόγραμμα που θα διαβάζει τις τιμές 2 ακέραιων μεταβλητών `x` και `y` και θα περιλαμβάνει τις συναρτήσεις `int MAX(int x, int y)` και `int MIN(int x, int y)`, οι οποίες υπολογίζουν το μέγιστο και τον ελάχιστο όρο μεταξύ δυο ακεραίων. Οι τιμές των συναρτήσεων θα εμφανίζονται με την παραπάνω σειρά από το κυρίως πρόγραμμα.

7. Να γραφεί ένα πρόγραμμα που θα διαβάζει τις τιμές 2 ακέραιων μεταβλητών `x`, `y` και θα περιέχει τις παρακάτω συναρτήσεις:

- `int add(int x, int y)`
- `int sub(int x, int y)`
- `int mult(int x, int y)`
- `int divd(int x, int y)` (αν ο παρονομαστής είναι 0 θα επιστρέφει την τιμή 0)

οι οποίες θα υπολογίζουν αντίστοιχα το άθροισμα, τη διαφορά, το γινόμενο και το ακέραιο πηλίκο δύο ακέραιων αριθμών. Τέλος να υπολογίζονται οι τιμές των:

- `mult(add(x,y), sub(y, divd(x,y))` και
- `divd(sub(mult(x, y), x), add(x, y))`.

Το κυρίως πρόγραμμα θα εμφανίζει τις τιμές των παραπάνω συναρτήσεων με αντίστοιχη σειρά. Δίνονται 2 ενδεικτικά στιγμιότυπα εκτέλεσης:

```
Dwse ton 1o arithmo: 6
Dwse ton 2o arithmo: 3
add: 9
sub: 3
mult: 18
divd: 2
exp1: 9
exp2: 1
```

```
Dwse ton 1o arithmo: 14
Dwse ton 2o arithmo: 25
add: 39
sub: -11
mult: 350
divd: 0
exp1: 975
exp2: 8
```

8. Να γίνει πρόγραμμα που θα διαβάζει τις τιμές 2 ακέραιων μεταβλητών και θα περιλαμβάνει τις συναρτήσεις:

- `int MKD (int x, int y)` και
- `int EKP (int x, int y)`,

οι οποίες θα υπολογίζουν το μέγιστο κοινό διαιρέτη (ΜΚΔ) και το ελάχιστο κοινό πολλαπλάσιο (ΕΚΠ) δύο ακέραιων αριθμών. Οι συναρτήσεις να γραφούν με τη βοήθεια των ασκήσεων 6 και 7. Για τον υπολογισμό του ΜΚΔ στηριχθείτε στις παρακάτω συνθήκες:

* ο `ΜΚΔ` των `Α` και `0` είναι `Α`
* αν `Α>Β` τότε ο `ΜΚΔ(Α, Β) = ΜΚΔ(Α-Β, Β)`

Στη συνέχεια το κυρίως πρόγραμμα θα εμφανίζει τις τιμές των παραπάνω συναρτήσεων με αντίστοιχη σειρά.

9. Γράψτε ένα πρόγραμμα καθοδηγούμενο από μενού επιλογών (menu-driven program) το οποίο μετατρέπει διάφορα μεγέθη από ένα είδος μονάδων σε άλλο, με τη βοήθεια συναρτήσεων:

- συνάρτηση που δέχεται λεπτά (`double`) και επιστρέφει ώρες (`double`)
- συνάρτηση που δέχεται πόδια (`double`) και επιστρέφει μέτρα (`double`) (1 πόδι = 0.3048 μέτρα)
- συνάρτηση που δέχεται βαθμούς Celsius (`double`) και επιστρέφει βαθμούς Fahrenheit (`double`) `(C=5/9*(F-32))`
- συνάρτηση που δέχεται βαθμούς Fahrenheit (`double`) και επιστρέφει βαθμούς Celsius (`double`).

Για την εμφάνιση του παρακάτω μενού επιλογών και το διάβασμα της επιλογής του χρήστη θα χρησιμοποιηθεί συνάρτηση:

```
AVAILABLE OPTIONS:
1. CONVERT MINUTES TO HOURS
2. CONVERT FEET TO METERS
3. CONVERT DEGREES CELSIUS TO DEGREES FAHRENHEIT
4. CONVERT DEGREES FAHRENHEIT TO DEGREES CELSIUS
5. QUIT
```

Όταν ο χρήστης επιλέξει μια από τις επιλογές του μενού στη συνέχεια θα δίνει την αντίστοιχη τιμή για μετατροπή. Το αποτέλεσμα της μετατροπής θα εμφανίζεται στο κυρίως πρόγραμμα και όχι στη συνάρτηση.

Πατώντας ENTER θα επαναλαμβάνεται το αρχικό μενού ώστε να συνεχίσει ο χρήστης να καλεί νέα διαδικασία.

10. Γράψτε πρόγραμμα που περιλαμβάνει μια συνάρτηση με το όνομα Valid_Date η οποία θα δέχεται 3 ακέραιους αριθμούς που αντιστοιχούν σε ημέρα, μήνα και έτος μιας ημερομηνίας και θα επιστρέφει την τιμή TRUE ή FALSE ανάλογα με το αν η ημερομηνία αυτή είναι αποδεκτή ή όχι (αν δηλαδή υπάρχει ή όχι στο ημερολόγιο). Θα πρέπει να ληφθούν υπόψη και τα δίσεκτα έτη (μπορεί να χρησιμοποιηθεί η συνάρτηση IsLeapYear που υπάρχει στις διαφάνειες του μαθήματος). Τα δεδομένα (ημέρα, μήνας και έτος) θα διαβάζονται στο κυρίως πρόγραμμα. Στο τέλος το πρόγραμμα θα εμφανίζει την τιμή TRUE ή FALSE.

11. Γράψτε ένα πρόγραμμα το οποίο θα δέχεται έναν αριθμό (τύπου int) που αντιστοιχεί στο τρέχον έτος. Στη συνέχεια, θα διαβάζει αγνώστου πλήθους τετράδες δεδομένων. Κάθε τετράδα θα περιλαμβάνει τα εξής:

- τρεις ακέραιους αριθμούς (τύπου int) που θα αντιστοιχούν σε μια ημερομηνία (ημέρα, μήνας, έτος, το έτος θα δίνεται ως τετραψήφιος, π.χ. 1999)
- έναν αριθμό που αντιστοιχεί σε ένα ποσό ΦΠΑ (long)

Οι ημερομηνίες πρέπει να ελέγχονται αν είναι αποδεκτές με τη συνάρτηση Valid_Date (βλ. άσκηση 10) και να ανήκουν οπωσδήποτε στο ίδιο έτος. Αν μια ημερομηνία δεν είναι αποδεκτή ή δεν ανήκει στο τρέχον έτος, τότε το αντίστοιχο ποσό ΦΠΑ δεν θα υπολογίζεται. Η εισαγωγή δεδομένων θα συνεχίζεται μέχρι να δοθεί η τιμή φρουρός −1 για τον 1ο ακέραιο της τετράδας δεδομένων – δηλαδή για τον ακέραιο αριθμό που αντιστοιχεί στην ημέρα (στην περίπτωση αυτή το πρόγραμμα δεν θα εμφανίζει στο χρήστη μηνύματα για το διάβασμα των υπόλοιπων τιμών).
Το τρέχον έτος θα διαβάζεται στο κυρίως πρόγραμμα, ενώ οι τετράδες των δεδομένων και ο υπολογισμός του ΦΠΑ θα γίνεται σε μια συνάρτηση Read_and_Calculate που θα δέχεται το τρέχον έτος και θα επιστρέφει το συνολικό ποσό ΦΠΑ. Το συνολικό ποσό ΦΠΑ για όλο το έτος (long) θα εμφανίζεται από το κυρίως πρόγραμμα.

```
Current year: 2021
Day: 1
Month: 1
Year: 2021
FPA: 19800
Day: 15
Month: 31
Year: 2021
FPA: 100
Day: 10
Month: 2
Year: 2021
FPA: 1235
Day: 21
Month: 1
Year: 2023
FPA: 235
Day: 29
Month: 2
Year: 2021
FPA: 1236
Day: 25
Month: 4
Year: 2021
FPA: 1546
Day: 23
Month: 3
Year: 2004
FPA: 21897
Day: -1
SYNOLO FPA: 22581
```

12. Να γραφεί πρόγραμμα που θα περιλαμβάνει τα παρακάτω :

- μια συνάρτηση με όνομα Parag που θα δέχεται έναν ακέραιο αριθμό μικρότερο του 12 και θα επιστρέφει το παραγοντικό του (long) και
- μια συνάρτηση calculate_sinx που θα υπολογίζει την τιμή του sinx από τον τύπο :

$$sinx = x - x^3/3! + x^5/5! - x^7/7! + \ldots$$

Ο υπολογισμός θα σταματά όταν υπολογιστεί ο όρος $x^{11}/11!$ ή όταν η ακρίβεια του αποτελέσματος γίνει μικρότερη του 0.0001. Στη συνέχεια το κυρίως πρόγραμμα θα εμφανίζει το αποτέλεσμα σε δεξιά στοίχιση των 12 χαρακτήρων με ακρίβεια 10 δεκαδικών ψηφίων. Το x (double) θα διαβάζεται σε ακτίνια (rads) από το κυρίως πρόγραμμα. Για τον υπολογισμό του x^n μπορείτε να χρησιμοποιήσετε τη συνάρτηση pow της βιβλιοθήκης math. Η συνάρτηση calculate_sinx μετά τον υπολογισμό της τιμής του sinx θα εμφανίζει την τιμή του n (x^n). Δίνονται 3 ενδεικτικά στιγμιότυπα εκτέλεσης:

```
Dwse to x se aktinia: 0.5
n: 7
sinx: 0.4794255332
```

```
Dwse to x se aktinia: 2.1
n: 11
sinx: 0.8632069372
```

```
Dwse to x se aktinia: 4.7
n: 11
sinx: -1.0790998936
```

13. Η ακολουθία Fibonacci ορίζεται ως εξής: ο πρώτος της όρος είναι 0, ο δεύτερος 1 και από εκεί και πέρα κάθε όρος προκύπτει από το άθροισμα των δύο προηγούμενων όρων:

```
0 1 1 2 3 5 8 13 21 34 55...
```

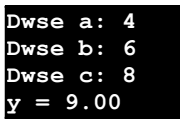
Να γράψετε μια συνάρτηση Fibonacci, η οποία θα δέχεται ένα θετικό ακέραιο `N` και θα επιστρέφει το Ν-οστό όρο της ακολουθίας Fibonacci. Στη συνέχεια, να γράψετε ένα πρόγραμμα στο οποίο θα διαβάζεται το `N` και χρησιμοποιώντας την παραπάνω συνάρτηση θα υπολογίζονται και εμφανίζονται οι `N` πρώτοι όροι της ακολουθίας με δεξιά στοίχιση 5 χαρακτήρων. Κάθε όρος της ακολουθίας θα εμφανίζεται σε διαφορετική γραμμή.

14. Να γίνει πρόγραμμα το οποίο:

- Θα διαβάζει τις τιμές τριών ακεραίων αριθμών `a`, `b`, `c`,
- Θα υπολογίζει και θα εμφανίζει την τιμή της παράστασης (double):

$$y=\frac{2*\max(a,b)+3*gr(a,b,c)}{4}$$

όπου `max(a, b)` είναι η συνάρτηση υπολογισμού του μεγίστου των αριθμών `a` και `b` και `gr(a, b, c)` η συνάρτηση υπολογισμού του μεγίστου τριών αριθμών `a`, `b`, `c`. Το διάβασμα των τριών αριθμών, ο υπολογισμός και εμφάνιση της τιμής της παράστασης θα γίνουν στο κυρίως πρόγραμμα, χρησιμοποιώντας τις συναρτήσεις `max` και `gr` που θα υλοποιήσετε ως συναρτήσεις στην C. Το αποτέλεσμα (y) θα εμφανίζεται με ακρίβεια 2 δεκαδικών ψηφίων.



15. Να γίνει πρόγραμμα το οποίο θα δέχεται ένα ποσό χρημάτων σε € και θα εμφανίζει την αξία των χρημάτων που θα έχουμε σε ένα από τα σημαντικότερα παγκοσμίως νομίσματα (δολάριο, λίρα, φράγκο Ελβετίας, δολάριο Καναδά ή γεν). Δίνονται οι ισοτιμίες:

1 Δολάριο	0,89 €
1 Λίρα	0,618 €
1 Φράγκο Ελβετίας	1,5465 €
1 Δολάριο Καναδά	1,3565 €
1 Γιεν	109,22 €

Το πρόγραμμα θα ελέγχεται από το παρακάτω μενού επιλογής και θα σταματάει όταν ο χρήστης επιλέξει από το μενού την επιλογή έξοδο.

```
1. Metetroph se dollaria
2. Metatroph se lires
3. Metatroph se fraga Elbetias
4. Metatroph se dollaria Canada
5. Metatroph se gien
6. Exodos
```

Το πρόγραμμα να αναπτυχθεί με τη χρήση:

1. συνάρτησης για την εμφάνιση του μενού και το διάβασμα της επιλογής (`int`) του χρήστη, η οποία θα επιστρέφεται από τη συνάρτηση
2. συνάρτησης που θα δέχεται ως είσοδο την επιλογή του χρήστη και το ποσό σε € (`double`) και θα επιστρέφει την αξία του (`double`) στο νόμισμα που επέλεξε ο χρήστης.

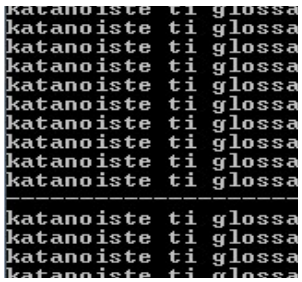
Η εμφάνιση της αξίας (του ποσού που έδωσε ο χρήστης σε €) στο νόμισμα που επέλεξε ο χρήστης θα γίνεται από το κυρίως πρόγραμμα, χρησιμοποιώντας ακρίβεια 2 δεκαδικών ψηφίων.

16. Να γραφεί συνάρτηση που θα δέχεται τρεις παραμέτρους και
-Αν η τιμή της πρώτης παραμέτρου είναι 1, να επιστρέφει ως τιμή το άθροισμα των δύο άλλων παραμέτρων.
-Αν η τιμή της πρώτης παραμέτρου είναι 2, να επιστρέφει ως τιμή το γινόμενο των άλλων δυο παραμέτρων.

-Αν η τιμή της πρώτης παραμέτρου είναι 3, να επιστρέφει ως τιμή τον μέσο όρο των άλλων δυο παραμέτρων.
-Αν η τιμή της πρώτης παραμέτρου δεν είναι ούτε 1, ούτε 2, ούτε 3, να εμφανίζει στην οθόνη μήνυμα λάθους "Αντικανονική κλήση συνάρτησης" και να σταματάει το πρόγραμμα με κωδικό εξόδου 1 (θα καλέσετε τη συνάρτηση `exit(1)` για να τερματιστεί η εκτέλεση του προγράμματος).
Να επιλέξετε εσείς τον τύπο της συνάρτησης και τον τύπο των παραμέτρων της. Να αναπτυχθεί πρόγραμμα στο οποίο η συνάρτηση main() θα καλεί την συνάρτηση που αναπτύξατε και θα εμφανίζει τα αποτελέσματά της για τιμές δεύτερης και τρίτης παραμέτρου 5 και 10, και τιμές πρώτης παραμέτρου 1, 2, 3 και 10. Τα αποτελέσματα των τεσσάρων κλήσεων της συνάρτησης φαίνονται παρακάτω.

17. Να γραφεί συνάρτηση με όνομα `total()` που να δέχεται ως παράμετρο έναν αριθμό και να επιστρέφει ως τιμή το άθροισμα των αριθμών από το 1 μέχρι την τιμή της παραμέτρου. Για παράδειγμα, η `total(100)` να επιστρέφει ως τιμή το άθροισμα των αριθμών από το 1 μέχρι το 100 (5050). Να αναπτυχθεί πρόγραμμα στο οποίο η συνάρτηση main() θα καλεί την total και θα εμφανίζει τα αποτελέσματά της για τιμές 100 και 1000, όπως φαίνεται στο παράδειγμα.

18. Να γραφεί συνάρτηση η οποία θα εμφανίζει δέκα φορές τη φράση "Κατανοήστε τη γλώσσα C". Επίσης, να γραφεί και μια δεύτερη έκδοση της ίδιας συνάρτησης η οποία θα εμφανίσει την ίδια φράση, αλλά το πλήθος των φορών θα της μεταβιβάζεται ως παράμετρος. (Οι συναρτήσεις θα δηλωθούν τύπου void δεδομένου ότι δεν επιστρέφουν τιμή.). Να αναπτυχθεί πρόγραμμα στο οποίο η συνάρτηση main() θα καλεί διαδοχικά τις δύο συναρτήσεις και θα εμφανίζει τα αποτελέσματά τους, όπως φαίνεται στο παράδειγμα.



19. Ο ακόλουθος τύπος του John Wallis (1616-1703) υπολογίζει προσεγγιστικά το π (3,14...). Να γραφεί πρόγραμμα το οποίο να χρησιμοποιεί αυτή τη σχέση και να υπολογίζει την τιμή του π. Το πρόγραμμα να χρησιμοποιεί συνάρτηση για τον υπολογισμό του δεξιού μέλους της σχέσης, η οποία θα δέχεται μια παράμετρο, που εκφράζει τη μέγιστη τιμή του x, όπως φαίνεται στον τύπο. Το αποτέλεσμα του προγράμματος για τιμή παραμέτρου 1000, φαίνεται παρακάτω. (Μπορείτε να χρησιμοποιήσετε την συνάρτηση pow για να υψώσετε στο τετράγωνο και η βιβλιοθήκη <math.h>).

$$\frac{\pi}{2}=\frac{2^2}{1*3}*\frac{4^2}{3*5}*\frac{6^2}{5*7}*\ldots*\frac{x^2}{(x-1)*(x+1)} \qquad \text{για } x \text{ άρτια } (x=2n)$$

20. Να γραφεί πρόγραμμα το οποίο να ζητάει δύο χρονολογίες (π.χ. 2000, 2010) και να εμφανίζει τα δίσεκτα έτη ανάμεσα σε αυτές τις χρονολογίες. Δίσεκτο είναι ένα έτος όταν διαιρείται ακριβώς με το 4. Όμως τα έτη που διαιρούνται ακριβώς με το 100 δεν είναι δίσεκτα εκτός αν διαιρούνται ακριβώς με το 400. Για την απόφαση αν ένα έτος είναι δίσεκτο ή όχι θα πρέπει να χρησιμοποιηθεί συνάρτηση στην οποία θα περνάει ως παράμετρος η χρονολογία και θα επιστρέφει τιμή 1 αν το έτος είναι δίσεκτο, και τιμή 0 αν δεν είναι.

21. Ο Δείκτης Μάζας Σώματος (ΔΜΣ) υπολογίζεται από τον τύπο B/Y^2 που Β το βάρος σε κιλά και Υ το ύψος σε μέτρα. Ανάλογα με την τιμή του ΔΜΣ ένα άτομο χαρακτηρίζεται σύμφωνα με τον παρακάτω πίνακα:

ΔΜΣ	**Περιγραφή**
μικρότερος από 18.5	Λιποβαρής
από 18.5 και μικρότερος του 25	Κανονικός
από 25 και μικρότερος του 30	Υπέρβαρος
από 30 και πάνω	Παχύσαρκος

Να γραφεί πρόγραμμα το οποίο θα ζητάει να πληκτρολογούμε το (double) βάρος και το (double) ύψος διάφορων ατόμων, θα υπολογίζει τον ΔΜΣ και θα εμφανίζει το χαρακτηρισμό του ατόμου (π.χ. υπέρβαρος). Το πρόγραμμα θα σταματάει όταν δοθεί τιμή 0 είτε για το βάρος είτε για το ύψος. Ο υπολογισμός του ΔΜΣ πρέπει να υλοποιείται από μια συνάρτηση. Επίσης από μια δεύτερη συνάρτηση θα πρέπει να υλοποιείται η εμφάνιση των αποτελεσμάτων.

22. Ο ΦΠΑ ενός προϊόντος μπορεί να ανήκει στις παρακάτω κατηγορίες:

Κατηγορία	**Ποσοστό ΦΠΑ**
1	**0.00**
2	**0.06**
3	**0.13**
4	**0.19**

Να γραφεί πρόγραμμα το οποίο θα ζητάει να πληκτρολογούμε το (int) πλήθος, τη (double) τιμή μονάδας και τη (int) κατηγορία ΦΠΑ για 5 προϊόντα. Το πρόγραμμα θα πρέπει να εμφανίζει το συνολικό κόστος της δαπάνης καθώς και το σύνολο του ΦΠΑ για όλα τα προϊόντα που αγοράσαμε. Ο υπολογισμός του ΦΠΑ πρέπει να υλοποιείται από μια συνάρτηση στην οποία θα μεταβιβάζεται το συνολικό ποσό ανά προϊόν καθώς και η κατηγορία στην οποία ανήκει. Αν ο χρήστης δώσει λάθος κατηγορία ΦΠΑ, τότε η *συνάρτηση* θα εμφανίσει μήνυμα λάθους, και θα επιστρέφει τιμή 0. Σε αυτή την περίπτωση, στο συνολικό κόστος δαπάνης υπολογίζεται κανονικά το κόστος του προϊόντος, το οποίο εισάχθηκε με λάθος ΦΠΑ, όπως φαίνεται στην δεξιά εικόνα παρακάτω.

```
Dwse to plithos temaxiwn apo to proion 1: 2
Dwse  timi  gia to proion 1: 10
Dwse  katigoria FPA gia to proion 1: 1
Dwse to plithos temaxiwn apo to proion 2: 4
Dwse  timi  gia to proion 2: 15
Dwse  katigoria FPA gia to proion 2: 4
Dwse to plithos temaxiwn apo to proion 3: 10
Dwse  timi  gia to proion 3: 20
Dwse  katigoria FPA gia to proion 3: 3
Dwse to plithos temaxiwn apo to proion 4: 1
Dwse  timi  gia to proion 4: 20
Dwse  katigoria FPA gia to proion 4: 2
Dwse to plithos temaxiwn apo to proion 5: 3
Dwse  timi  gia to proion 5: 100
Dwse  katigoria FPA gia to proion 5: 1
Synoliko kostos: 638.60
Synoliko fpa: 38.60
```

```
Dose to plithos temaxiwn apo to proion 1: 1
Dose  timi  gia to proion 1: 1
Dose  katigoria FPA gia to proion 1: 0
Lathos kathgoria FPA
Dose to plithos temaxiwn apo to proion 2: 1
Dose  timi  gia to proion 2: 1
Dose  katigoria FPA gia to proion 2: 2
Dose to plithos temaxiwn apo to proion 3: 1
Dose  timi  gia to proion 3: 1
Dose  katigoria FPA gia to proion 3: 2
Dose to plithos temaxiwn apo to proion 4: 1
Dose  timi  gia to proion 4: 1
Dose  katigoria FPA gia to proion 4: 2
Dose to plithos temaxiwn apo to proion 5: 1
Dose  timi  gia to proion 5: 1
Dose  katigoria FPA gia to proion 5: 2
Synolikko kostos: 5.24
Synoliko fpa: 0.24
```

23. Γράψτε πρόγραμμα που περιλαμβάνει μια συνάρτηση με το όνομα **`Valid_Time`** η οποία θα δέχεται ως τυπικές παραμέτρους 3 ακέραιους αριθμούς (int) που αντιστοιχούν σε ώρες, λεπτά και δευτερόλεπτα μιας χρονικής στιγμής της ημέρας (ώρα) και θα επιστρέφει την τιμή TRUE ή FALSE ανάλογα με το αν η δοθείσα χρονική στιγμή (ώρα) είναι έγκυρη ή όχι (αν δηλαδή οι ώρες θα πρέπει να είναι από 0 έως και 23 και τα λεπτά και δευτερόλεπτα να είναι από 0 έως και 59). Τα δεδομένα (ώρες, λεπτά και δευτερόλεπτα) θα ζητούνται από τον χρήστη στο *κυρίως πρόγραμμα*, όπως φαίνεται στα ακόλουθα παραδείγματα. Στο τέλος το πρόγραμμα θα εμφανίζει το μήνυμα "Valid: yes" ή "Valid: no" από το *κυρίως πρόγραμμα* (συνάρτηση **`main()`**, δηλαδή η εντολή εμφάνισης του μηνύματος (**`printf`**) θα είναι στην **`main`**).

Παραδείγματα εκτέλεσης:
Εκτέλεση 1:

```
Dwse tis ores: 16
Dwse ta lepta: 30
Dwse ta defterolepta: 30
Valid: yes
```

Εκτέλεση 2:

```
Dwse tis ores: 23
Dwse ta lepta: 60
Dwse ta defterolepta: 0
Valid: no
```

24. Ένα Internet café χρεώνει με την εξής διαδικασία:
Ελάχιστη χρέωση: 2€ για κάθε παιχνίδι που αντιστοιχεί σε παραμονή του χρήστη μέχρι 1 ώρα. Για κάθε επιπλέον ώρα η χρέωση είναι 0.5€ Το μέγιστο της χρέωσης δεν μπορεί πάντως να ξεπερνάει τα 10€. Γράψτε ένα πρόγραμμα που να περιλαμβάνει μια συνάρτηση calc_charge η οποία θα δέχεται τη διάρκεια παραμονής σε ώρες και θα επιστρέφει τη χρέωση σε ευρώ. Η διάρκεια παραμονής θα διαβάζεται στο κυρίως πρόγραμμα το οποίο και θα εμφανίζει τη χρέωση.

```
Dose ores paramonis: 7
H xreosi einai 5 euro
```

25. Να αναπτύξετε μία συνάρτηση με όνομα `max(a,b)`, που να επιστρέφει τον μέγιστο από τους δύο ακεραίους a,b. Οι ακέραιοι εισάγονται από τον χρήστη όπως φαίνεται στο ακόλουθο παράδειγμα εκτέλεσης.

```
Dose x: 23
Dose y: 17
Max: 23
Press any key to continue . . .
```

26. Να δημιουργήσετε τις συναρτήσεις `f()` και `g()`, οι οποίες ορίζονται από τους παρακάτω μαθηματικούς τύπους, οι οποίες θα πρέπει να υλοποιηθούν ως συναρτήσεις στην γλώσσα C:

$$f(x)=\begin{cases} x+2\ , & x>0 \\ -3x+7\ , & x\le 0 \end{cases} \qquad g(x,y)=\begin{cases} x-y, & x>0 \ \ και \ \ y>0 \\ 7x-5, & αλλιώς \end{cases}$$

Οι τιμές των παραμέτρων **x,y** στις παραπάνω συναρτήσεις είναι ακέραιες (int). Ο τύπος των συναρτήσεων θα καθοριστεί από εσάς. Να υλοποιηθεί πρόγραμμα το οποίο ζητά από τον χρήστη τις παραμέτρους x και y, και έπειτα εμφανίζει (α) το αποτέλεσμα της συνάρτησης **f(x)**, (β) το αποτέλεσμα της συνάρτησης **g(x,y)** και το αποτέλεσμα της συνάρτησης **f(g(x))**, όπως φαίνονται στα ακόλουθα παραδείγματα εκτέλεσης. Η εμφάνιση των αποτελεσμάτων θα γίνεται μέσω του κυρίως προγράμματος (συνάρτηση **main()**, δηλαδή η εντολή εμφάνισης των αποτελεσμάτων (**printf**) θα είναι στην **main()**).

Παραδείγματα εκτέλεσης:
Εκτέλεση 1:

```
Enter x: 2
Enter y: 1
f(x) = f(2) = 4
g(x,y) = g(2,1) = 1
f(g(x,y)) = f(g(2,1)) = 3
```

Εκτέλεση 2:

```
Enter x: 2
Enter y: -1
```

```
f(x) = f(2) = 4
g(x,y) = g(2,-1) = 9
f(g(x,y)) = f(g(2,-1)) = 11
```

Εκτέλεση 3:

```
Enter x: -1
Enter y: -2
f(x) = f(-1) = 10
g(x,y) = g(-1,-2) = -12
f(g(x,y)) = f(g(-1,-2)) = 43
```

27. Στις περιπτώσεις κωδικών που αποτελούνται από ένα μεγάλο αριθμό ψηφίων (πχ. λογαριασμών τραπεζών, κωδικών πληρωμής κλπ) για την ασφάλεια των συναλλαγών θα πρέπει να υπάρχει ένας εύκολος τρόπος εξασφάλισης ότι ο χρήστης δεν πληκτρολογεί λάθος τα ψηφία. Ο τρόπος είναι να προστίθενται στους κωδικούς *ψηφία ελέγχου*. Ένας απλός τρόπος για το παραπάνω είναι να προστίθενται 2 ψηφία ελέγχου στο τέλος του κωδικού δημιουργώντας έτσι το *τελικό κωδικό*. Για παράδειγμα:

Αρχικός κωδικός (N)	**Ψηφία Ελέγχου**	**Τελικός Κωδικός (EN)**
12558	60	1255860
55257	03	5525703

Τα *ψηφία ελέγχου* ενός αρχικού κωδικού N (long) δίνονται από το τύπο:

$$\textit{Ψηφία ελέγχου} = (98 - (N*100) \% 97) \% 97 \qquad (1)$$

Ο *έλεγχος εγκυρότητας* ενός *τελικού κωδικού EN* υπολογίζεται εύκολα ελέγχοντας αν η πράξη *EN % 97* έχει ως αποτέλεσμα 1, δηλαδή δεδομένου ενός τελικού κωδικού EN αυτός είναι έγκυρος αν το ακέραιο υπόλοιπο του με τον αριθμό 97 είναι ένα.

Να υλοποιήσετε:
(α) Μια συνάρτηση **`encode`** η οποία δέχεται *ένα αρχικό κωδικό* (long) και επιστρέφει τον αντίστοιχο *τελικό κωδικό* (long). Αναφέρεται και πάλι ότι ο *τελικός κωδικός* προκύπτει από την προσθήκη στο τέλος ενός *αρχικού κωδικού* των *ψηφίων ελέγχου,* τα οποία δίνονται από το τύπο (1).

(β) μια συνάρτηση **`check`** η οποία δέχεται *ένα τελικό κωδικό* και επιστρέφει TRUE αν είναι έγκυρος και FALSE σε αντίθετη περίπτωση. Αναφέρεται και πάλι ότι ο έλεγχος γίνεται εξετάζοντας αν το ακέραιο υπόλοιπο του *τελικού κωδικού* με το 97 είναι ίσο με 1.

(γ) ένα πρόγραμμα το οποίο δέχεται από τον χρήστη δύο όρια *αρχικών κωδικών* (long) και εμφανίζει για κάθε κωδικό ανάμεσα σε αυτά τα δύο όρια τον αρχικό κωδικό, τον τελικό κωδικό και αν είναι έγκυρος ή όχι, σύμφωνα με τα παραδείγματα που ακολουθούν. Η εμφάνιση των μηνυμάτων θα γίνεται μέσω του κυρίως προγράμματος (συνάρτηση **`main()`**, δηλαδή η εντολή εμφάνισης των μηνυμάτων (**`printf`**) θα είναι στην **`main()`**).
Υποθέστε ότι ο χρήστης δίνει πάντα όρια μεγαλύτερα του 0 και ότι το κάτω όριο (lower limit) θα είναι πάντα μικρότερο ή ίσο με το το άνω όριο (upper limit)). Δεν απαιτείται έλεγχος.
Σημείωση: Αν το πρόγραμμά σας είναι σωστό, τότε όλοι οι τελικοί κωδικοί που θα εμφανίζεται θα είναι έγκυροι.

Παραδείγματα Εκτέλεσης
Εκτέλεση 1:

```
Lower Limit:1
Upper Limit:5
Code: 1 Encoding: 195 isValid:yes
Code: 2 Encoding: 292 isValid:yes
Code: 3 Encoding: 389 isValid:yes
Code: 4 Encoding: 486 isValid:yes
Code: 5 Encoding: 583 isValid:yes
```

Εκτέλεση 2:

```
Lower Limit:12458
Upper Limit:12461
Code: 12458 Encoding: 1245869 isValid:yes
Code: 12459 Encoding: 1245966 isValid:yes
Code: 12460 Encoding: 1246063 isValid:yes
Code: 12461 Encoding: 1246160 isValid:yes
```

28. Να γράψετε πρόγραμμα που θα διαβάζει *τον κωδικό και την τιμή* ενός αγνώστου πλήθους προϊόντων που αγόρασε κάποιος σε ένα πολυκατάστημα. Ο κωδικός θα πρέπει να είναι στο διάστημα [0..2000] και να γίνεται ο αντίστοιχος έλεγχος από το πρόγραμμα. Η είσοδος των δεδομένων (*κωδικός και τιμή*) θα συνεχίζεται μέχρι να δοθεί ως κωδικός το μηδέν (δείτε παράδειγμα εκτέλεσης).

Έκπτωση
Στην περίοδο των εκπτώσεων γίνεται σε κάθε προϊόν *έκπτωση* ανάλογα με τον κωδικό του, ως εξής:

Κωδικός προϊόντος	Ποσοστό έκπτωσης
1..300	5%
301..500	10%
501..1000	15%
1001..2000	20%

Υπολογισμός Πόντων
Για κάθε προϊόν δίνονται κάποιοι πόντοι, τους οποίους ο πελάτης εξαργυρώνει με διάφορα δώρα. Συγκεκριμένα, για κάθε προϊόν δίνεται:

- ένας πόντος ανεξάρτητα από την τιμή του και
- επιπλέον για τα προϊόντα με κωδικό από 1000 έως και 1500 ένας πόντος για κάθε 5 Ευρώ της τελικής αξίας του προϊόντος. (δηλαδή της αξίας που προκύπτει μετά την έκπτωση).Οι επιπλέον πόντοι υπολογίζονται από τον τύπο

```
επιπλέον_πόντοι = ceil(τελική αξία/5)
```

Για παράδειγμα:
Για ένα προϊόν με κωδικό 1200 και τιμή 100€:

- η τελική τιμή θα είναι: 80€
- το ποσό της έκπτωσης: 20€
- οι πόντοι: 17

Για ένα προϊόν με κωδικό 100 και τιμή 10€:

- η τελική τιμή θα είναι: 9.50€
- το ποσό της έκπτωσης: 0.50€
- οι πόντοι: 1

Το πρόγραμμα θα εμφανίζει στην οθόνη τα ακόλουθα:

- το τελικό σύνολο που θα πληρώνει ο πελάτης (μετά την έκπτωση) (*με 2 δεκαδικά ψηφία*)
- το συνολικό ποσό έκπτωσης (*με 2 δεκαδικά ψηφία*)
- το σύνολο των πόντων

Παράδειγμα εκτέλεσης του προγράμματος.

```
Dwse ton kwdiko: 2005
Lathos Kwdikos (0-2000)
Dwse ton kwdiko: 1200
Dwse thn timh: 100
Dwse ton kwdiko: 231
Dwse thn timh: 10
Dwse ton kwdiko: 0
------------------
Teliko synolo: 89.50 Euro
Ekptwsh: 20.50 Euro
Pontoi: 18
```

Η υλοποίηση του προγράμματος *να βασιστεί στις ακόλουθες συναρτήσεις*:

- Συνάρτηση **`get_code()`** η οποία εμφανίζει τα προτρεπτικά μηνύματα εισαγωγής του κωδικού του προϊόντος, κάνει τον έλεγχο για έγκυρες τιμές και αν η τιμή κωδικού που εισάγει ο χρήστης είναι έγκυρη, τότε επιστρέφει τον κωδικό, αλλιώς εμφανίζει το μήνυμα λάθους "Lathos Kwdikos (0-2000)".
- Συνάρτηση **`discount_percentage()`** η οποία επιστρέφει το ποσοστό έκπτωσης, δεδομένου του κωδικού του προϊόντος.
- Συνάρτηση **`product_points()`** η οποία υπολογίζει και επιστρέφει τους πόντους που αντιστοιχούν σε ένα προϊόν, δεδομένων του κωδικού και της τιμής του προϊόντος.

29. Υλοποιείστε τις αντίστοιχες συναρτήσεις στην γλώσσα C των ακόλουθων μαθηματικών ορισμών:

- Ο *μέγιστος κοινός διαιρέτης (gdc)* δύο μη αρνητικών ακεραίων δίνεται από τον ακόλουθο αναδρομικό ορισμό:

$$gcd(a,b)=\begin{cases} a, & \text{εάν } b=0 \\ gcd(b, a\%b), & \text{αλλιώς} \end{cases}$$

- Δυο ακέραιοι αριθμοί λέγονται *σχετικά πρώτοι (coprime)* αν ο μέγιστος κοινός διαιρέτης τους είναι το 1.
- Η *συνάρτηση φ του Euler (phi)* ενός θετικού ακεραίου N είναι το πλήθος των θετικών ακεραίων αριθμών μικρότερων ή ίσων του N, οι οποίοι είναι σχετικά πρώτοι με το N (δηλαδή οι ακέραιοι K με *1 <= K <= N*).

Να υλοποιήσετε ένα πρόγραμμα σε C, το οποίο ζητά από τον χρήστη ένα ακέραιο X και εμφανίζει τις τιμές της συνάρτησης *φ* του *Euler* από το 1 μέχρι και τον αριθμό X. Θεωρείστε ότι ο χρήστης δίνει πάντα αριθμό μεγαλύτερο ή ίσο του 1 - δεν απαιτείται έλεγχος. Η εμφάνιση των μηνυμάτων θα γίνεται μέσω του κυρίως προγράμματος (συνάρτηση **`main()`**, δηλαδή η εντολή εμφάνισης των μηνυμάτων (**`printf`**) θα είναι στην **`main()`**).

Παραδείγματα Εκτέλεσης
Εκτέλεση 1:

```
Enter X:5
phi(1) = 1
phi(2) = 1
phi(3) = 2
phi(4) = 2
phi(5) = 4
```

Εκτέλεση 2:

```
Enter X:10
phi(1) = 1
phi(2) = 1
phi(3) = 2
phi(4) = 2
phi(5) = 4
phi(6) = 2
phi(7) = 6
phi(8) = 4
phi(9) = 6
phi(10) = 4
```

30. Να γραφεί συνάρτηση **calc**, η οποία να δέχεται τρεις ακέραιες παραμέτρους και η οποία έχει την ακόλουθη συμπεριφορά

- Αν η πρώτη παράμετρος είναι ίση με 1, τότε επιστρέφει το άθροισμα των δύο επόμενων παραμέτρων, για παράδειγμα η κλήση **calc(1, 3 ,4)** επιστρέφει 7**.**
- Αν η πρώτη παράμετρος είναι ίση με 2, τότε επιστρέφει τη διαφορά της τρίτης παραμέτρου από την δεύτερη, για παράδειγμα η κλήση **calc(1, 3 ,4)** επιστρέφει -1.

- Αν η πρώτη παράμετρος είναι ίση με 3, τότε επιστρέφει τον μέγιστο ακέραιο μικρότερο από τον μέσο όρο των δύο επόμενων παραμέτρων, για παράδειγμα η κλήση **calc(3, 3 ,4)** επιστρέφει 3 (χρήση της συνάρτηση floor(), της math.h στον υπολογισμό του μέσου όρου).
- Αν η πρώτη παράμετρος δεν είναι ίση με μια από τις παραπάνω τιμές 1, 2, 3 τότε επιστρέφει 0.

Να γραφεί πρόγραμμα το οποίο χρησιμοποιώντας την συνάρτηση **calc**,

- θα ζητά από τον χρήστη ένα θετικό ακέραιο **n** μεγαλύτερο του 1 (δεν χρειάζεται να γίνεται έλεγχος αν η τιμή είναι σωστή)
- θα τυπώνει το άθροισμα των αποτελεσμάτων της εφαρμογής των διαφορετικών *αριθμητικών πράξεων* που προσφέρονται από την συνάρτηση **calc** (δηλαδή 1,2,3) για όλους τους συνδυασμούς τιμών της δεύτερης και τις τρίτης παραμέτρου από 1 έως και **n**.

Για παράδειγμα αν ο χρήστης δώσει την τιμή **5**, τότε το πρόγραμμα θα υπολογίζει ουσιαστικά την παράσταση

calc(1,1,1)+calc(1,1,2)+calc(1,1,3)+calc(1,1,4)+...+ calc(3,5,5).

```
Dwse n: 5
Result: 219
```