Δ.Π.Θ. Πολυτεχνική Σχολή
Τμήμα Μηχανικών Παραγωγής & Διοίκησης
Τομέας Συστημάτων Παραγωγής
Αναπληρωτής Καθηγητής Στέφανος Δ. Κατσαβούνης

Ακαδ. Έτος: **2019-2020**

1 ΟΚΤ 2019

1ο ΦΥΛΛΑΔΙΟ ΑΣΚΗΣΕΩΝ
(επανάληψη στη γλώσσα C & εισαγωγή στη γλώσσα C++)

ΑΣΚΗΣΗ-1

Ένας ακέραιος αριθμός λέγεται τριμορφικός όταν τα τελευταία ψηφία του κύβου του αριθμού είναι ο ίδιος ο αριθμός π.χ. 49^3=117649, 25^3=15625 κλπ.
Να γραφεί πρόγραμμα σε γλώσσα C++ που θα βρίσκει, θα μετρά και θα εμφανίζει όλους τους τριμορφικούς αριθμούς που είναι μεγαλύτεροι του 9 και μικρότεροι του 100. Το πρόγραμμα θα εμφανίζει στο τέλος και το πλήθος αυτών των αριθμών.

ΑΣΚΗΣΗ-2

Να βρεθούν και να εμφανιστούν όλοι οι μονοψήφιοι και διψήφιοι θετικοί ακέραιοι αριθμοί που ικανοποιούν την παρακάτω ιδιότητα:
Το άθροισμα των ψηφίων του τετραγώνου του αριθμού για τους μονοψήφιους θετικούς ακέραιους αριθμούς ή το άθροισμα του αριθμού που προκύπτει από τα 2 τελευταία ψηφία του τετραγώνου του αριθμού και του αριθμού που προκύπτει από τα υπόλοιπα αριστερότερα ψηφία για τους διψήφιους να είναι ίσο με τον αριθμό. Παραδείγματα:

Μονοψήφιος : 9	9^2=81	8+1=9	ΕΙΝΑΙ
Διψήφιος :14	14^2=196	1+96=97	ΔΕΝ ΕΙΝΑΙ !
Διψήφιος : 45	45^2=2025	20+25=45	ΕΙΝΑΙ

Να γραφεί πρόγραμμα σε γλώσσα C++ που θα βρίσκει και θα εμφανίζει αυτούς τους αριθμούς. Στο τέλος θα πρέπει να εμφανίζεται και το πλήθος αυτών των αριθμών.

ΑΣΚΗΣΗ-3

Να γραφεί πρόγραμμα σε γλώσσα C++ που θα υπολογίζει τα παρακάτω αθροίσματα με ακρίβεια 10^{-4}. Η ακρίβεια υπολογισμού είναι η απόλυτη τιμή της διαφοράς δύο διαδοχικών τιμών του αθροίσματος.

1. $\frac{1}{3}-\frac{2}{5}+\frac{3}{7}-\frac{4}{9}+\ldots$
2. $\frac{1}{1+\sqrt{2}}-\frac{1}{\sqrt{2}+\sqrt{3}}+\frac{1}{\sqrt{3}+\sqrt{4}}-\frac{1}{\sqrt{4}+\sqrt{5}}+\ldots$
3. $ln\,x=(x-1)-\frac{1}{2}(x-1)^2+\frac{1}{3}(x-1)^3-\frac{1}{4}(x-1)^4+\ldots$
4. $1+\frac{x^2}{1\cdot 3}+\frac{x^4}{3\cdot 5}+\frac{x^6}{5\cdot 7}+\ldots+\frac{x^k}{m\cdot n}$
5. $1+x+\frac{x^2}{2!}+\frac{x^3}{3!}+\ldots+\frac{x^n}{n!}$

6. $$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \dots$$

7. $$e^{-x^2} = 1 - x^2 + \frac{x^4}{2!} - \frac{x^6}{3!} + \frac{x^8}{4!} - \dots$$

8. $$\sin^{-1} x = x + \left(\frac{1}{2}\right)\left(\frac{x^3}{3}\right) + \left(\frac{1}{2}\right)\left(\frac{3}{4}\right)\left(\frac{x^5}{5}\right) + \left(\frac{1}{2}\right)\left(\frac{3}{4}\right)\left(\frac{5}{6}\right)\left(\frac{x^7}{7}\right) + \dots$$

Η τιμή του x θα δίνεται από το πληκτρολόγιο (εντολή cin). Το πρόγραμμα, σε κάθε βήμα του, πρέπει να εμφανίζει όλες τις διαδοχικές τιμές του αθροίσματος καθώς και τη διαφορά τους μέχρι να επιτευχθεί η επιθυμητή ακρίβεια. Μετά τον υπολογισμό και την εμφάνιση του αποτελέσματος να εμφανίσετε το πλήθος των όρων που χρησιμοποιήθηκαν στον υπολογισμό.

ΑΣΚΗΣΗ-4

Χρησιμοποιώντας το παρακάτω τμήμα κώδικα σε γλώσσα C, που βρίσκει αν ένας ακέραιος και θετικός αριθμός $k \geq 2$ είναι πρώτος (prime) αριθμός, να δημιουργήσετε μια συνάρτηση σε γλώσσα C++, με όνομα is_prime, που θα δέχεται ως όρισμα έναν θετικό ακέραιο αριθμό $k \geq 2$ και θα επιστρέφει αν ο αριθμός είναι πρώτος ή όχι.

```
i=2; flag=0;
while ((i<=k/2) && (flag==0))
{
      if (k%i==0) flag=1;
      i++;
}
if (flag==0) printf("number %4d  is prime \n",k);
```

Στη συνέχεια, να γραφεί πρόγραμμα που θα καλεί τη συνάρτηση. Το πρόγραμμα θα δημιουργεί, μέσω της συνάρτησης rand( ) έναν θετικό ακέραιο αριθμό στην περιοχή [1, 9999] και θα καλεί τη συνάρτηση is_prime για να εμφανίζει τον αριθμό και ένα μήνυμα, αν ο αριθμός αυτός είναι πρώτος (prime) ή όχι.

ΑΣΚΗΣΗ-5

Χρησιμοποιώντας τη συνάρτηση is_prime, της άσκησης 4, να γράψετε τα αντίστοιχα προγράμματα σε γλώσσα C++ για τα παρακάτω προβλήματα που σχετίζονται με πρώτους (primes) αριθμούς:

1. Ορισμένοι πρώτοι αριθμοί (prime numbers), έστω p_n , ονομάζονται good primes εφόσον ικανοποιούν τη συνθήκη :

$$p_n^2 > p_{n-1} \cdot p_{n+1}$$

Μερικοί τέτοιοι πρώτοι αριθμοί είναι οι : 5, 11, 17, 29, 37, 41, 53, ... (δηλ. πχ $5^2 = 25 > 4 \cdot 6 = 24$)
Να γραφεί πρόγραμμα σε γλώσσα C++ που θα βρίσκει και θα εμφανίζει όλους αυτούς τους πρώτους αριθμούς που είναι μικρότεροι του 1000.

2. Ένας παλινδρομικός πρώτος (prime) αριθμός είναι ένας ακέραιος θετικός αριθμός που είναι ταυτόχρονα παλινδρομικός και πρώτος (prime). Ορισμένοι τέτοιοι αριθμοί είναι οι : 2, 3, 5, 7, 11, 101, 131, 151, 181, 191, 313, 353, 373, 383, 727, 757, 787, ...

Να γραφεί αλγόριθμος ή πρόγραμμα σε γλώσσα C++ που θα βρίσκει και θα εμφανίζει όλους τους παλινδρομικούς πρώτους αριθμούς, εφόσον αυτοί είναι μικρότεροι του 10000. Στο τέλος να εμφανίσετε και το πλήθος αυτών των αριθμών.

---

3. Οι sexy πρώτοι αριθμοί είναι ζεύγη πρώτων (primes) αριθμών της μορφής

$$(p, p+6)$$

και ονομάζονται έτσι επειδή η λατινική λέξη για τον αριθμό 6 είναι "sex". Τα πρώτα ζεύγη των sexy primes είναι (5, 11), (7, 13), (11, 17), (13, 19), (17, 23), (23, 29), (31, 37), (37, 43), (41, 47), (47, 53), ...

Να γραφεί πρόγραμμα σε γλώσσα C++ που θα βρίσκει και θα εμφανίζει όλα τα sexy ζεύγη πρώτων αριθμών, ένα σε κάθε σειρά εμφάνισης, εφόσον και οι δύο αριθμοί του ζεύγους είναι μικρότεροι του 1000. Στο τέλος να εμφανίσετε και το πλήθος αυτών των ζευγών.

---

4. Ορισμένοι πρώτοι αριθμοί (prime numbers) ονομάζονται παλινδρομικοί πρώτοι αριθμοί αν ο ανάστροφός τους είναι επίσης πρώτος (prime) . Π.χ. ο αριθμός 13 και ο αριθμός 31 είναι παλινδρομικοί πρώτοι αριθμοί διότι και οι ανάστροφοί τους (δηλ. οι 31 και 13 αντίστοιχα) είναι πρώτοι (primes).

Μερικοί τέτοιοι πρώτοι αριθμοί είναι οι : 13, 17, 31, 37, 71, 73, 79, 97, 107, 113, ...

Να γραφεί πρόγραμμα σε γλώσσα C++ που θα βρίσκει και θα εμφανίζει όλους αυτούς τους πρώτους αριθμούς που είναι μικρότεροι του 1000. Στο τέλος να εμφανίσετε και το πλήθος αυτών των αριθμών.

---

5. Ένας πρώτος (prime) αριθμός ονομάζεται Mersenne prime εάν μπορεί να γραφεί στη μορφή $2^p - 1$, όπου ο p είναι κάποιος θετικός ακέραιος αριθμός. Η σειρά των Mersenne primes ξεκινά με τους αριθμούς 3, 7, 31, ...

p	$2^p - 1$
2	3
3	7
5	31
....	

Να γραφεί πρόγραμμα σε γλώσσα C++ που θα βρίσκει και θα εμφανίζει όλους αυτούς τους πρώτους αριθμούς που είναι μικρότεροι του 2000.

---

6. Ορισμένοι πρώτοι αριθμοί μπορούν να παραχθούν από την ακόλουθη σχέση:

$$n^3 - (n-1)^3, \quad n = 1, 2, 3, \ldots$$

Μερικοί τέτοιοι πρώτοι αριθμοί είναι οι : 7, 19, 37, 61, 127, 271, ...

Να γραφεί πρόγραμμα σε γλώσσα C++ που θα βρίσκει και θα εμφανίζει όλους αυτούς τους πρώτους αριθμούς που είναι μικρότεροι του 1000.

ΑΣΚΗΣΗ-6

Να γραφεί μια συνάρτηση σε γλώσσσα C++, με όνομα find_primes που θα δέχεται ως είσοδο :

- δύο μονοδιάστατους πίνακες ακεραίων θετικών αριθμών, έστω A και B στο διάστημα [0 – 499] με μέγεθος *N* (δηλ. και οι δύο πίνακες έχουν το ίδιο πλήθος στοιχείων)

Η συνάρτηση θα επιστρέφει στη συνάρτηση main( ) :

Ένα νέο πίνακα, έστω C , που θα περιλαμβάνει μόνον τα στοιχεία των πινάκων A και B που είναι πρώτοι αριθμοί (prime numbers) , σε αύξουσα διάταξη, χωρίς να χρησιμοποιηθεί διαδικασία ταξινόμησης. Κάθε στοιχείο θα πρέπει να εμφανίζεται μόνον μία φορά. Για τον έλεγχο αν ένας αριθμός είναι πρώτος ή όχι πρέπει να χρησιμοποιηθεί η συνάρτηση is_prime της άσκησης 4.

Στη συνέχεια να γραφεί ένα πρόγραμμα σε γλώσσα C++ που :

1. θα γεμίζει δύο μονοδιάστατους πίνακες N θέσεων (N =γνωστό, σταθερά που δηλώνεται αρχικά) με ακέραιους και θετικούς αριθμούς στην περιοχή [0 , 499]. Αν η εισαγωγή τιμών γίνει με χρήση της scanf θα πρέπει να υπάρχουν οι κατάλληλοι έλεγχοι εγκυρότητας τιμών. (δηλ. οι αριθμοί να ανήκουν στο διάστημα [0, 499]) , ωστόσο είναι προτιμότερη η χρήση της συνάρτησης rand( ).
2. θα καλεί τη συνάρτηση find_primes
3. θα εμφανίζει τα αποτελέσματα που θα επιστρέφει η συνάρτηση

ΠΑΡΑΔΕΙΓΜΑ με N=10

Πίνακας A :

7	30	**197**	99	4	**19**	**211**	**271**	478	**79**

Πίνακας B :

22	**271**	**197**	88	**337**	402	**157**	200	413	175

Πίνακας C :

7	19	79	157	197	211	271	337

ΑΣΚΗΣΗ-7

Η ηλεκτρική αντίσταση μιας αντίστασης καθορίζεται από το χρώμα της. Τα χρώματα που χρησιμοποιούνται και οι ακέραιοι που αντιστοιχούν σε κάθε μία δίνονται στον ακόλουθο πίνακα:

COLOR	*CODE*	*COLOR*	*CODE*
Silver	-2	Yellow	4
Gold	-1	Green	5
Black	0	Blue	6
Brown	1	Violet	7
Red	2	Gray	8
Orange	3	White	9

Αν οι κωδικοί τριών αντιστάσεων είναι κατά σειρά c_1, c_2 και c_3 τότε η συνολική αντίσταση σε Ohms είναι $(10 \cdot c_1 + c_2) \cdot 10^{c_3}$.

Να γραφεί ένα πρόγραμμα που θα εμφανίζει στην οθόνη μια λίστα των χρωμάτων και θα περιμένει από το χρήστη να εισάγει ακριβώς τρία χρώματα αντιστάσεων (που θα περιλαμβάνει ένα καλώδιο), επιλέγοντας ένα γράμμα για κάθε χρώμα (π.χ. B για Black, S για Silver, N για Brown κλπ). Στη συνέχεια θα υπολογίζει και θα εμφανίζει τη συνολική αντίσταση του καλωδίου.

Η εμφάνιση της λίστας των χρωμάτων θα γίνεται μέσω μιας συνάρτησης με όνομα print_codes. Η συνάρτηση decode_char θα δέχεται ως παράμετρο το χαρακτήρα που αντιστοιχεί σε ένα χρώμα, θα μετατρέπει το χαρακτήρα στην κατάλληλη αριθμητική τιμή και θα επιστρέφει αυτήν την αριθμητική τιμή. Όλοι οι άλλοι υπολογισμοί θα γίνονται στη συνάρτηση main( ).

ΑΣΚΗΣΗ-8

Μια βασική Πυθαγόρεια τριάδα ορίζεται από τρεις ακεραίους αριθμούς i, j, k τέτοιους ώστε $i^2 + j^2 = k^2$ και οι i, j, k δεν πρέπει να έχουν κοινούς διαιρέτες εκτός από το 1. Π.χ. οι ακέραιοι 3,4,5 αποτελούν μια βασική Πυθαγόρεια τριάδα. Ωστόσο οι 6,8,10 αν και $6^2 + 8^2 = 10^2$ δεν αποτελούν μια βασική Πυθαγόρεια τριάδα επειδή και οι τρεις διαιρούνται με το 2. Να γράψετε ένα πρόγραμμα σε γλώσσα C++ που θα βρίσκει και θα εμφανίζει στην παρακάτω μορφή όλες τις βασικές Πυθαγόρειες τριάδες με $i, j, k \le 1000$.

```
Maximum integer to test = 1000
Triple #  1: 3*3 + 4*4 = 5*5 = 25
Triple #  2: 5*5 + 12*12 = 13*13 = 169
Triple #  3: 8*8 + 15*15 = 17*17 = 289
Triple #  4: 7*7 + 24*24 = 25*25 = 625
Triple #  5: 20*20 + 21*21 = 29*29 = 841
Triple #  6: 12*12 + 35*35 = 37*37 = 1369
Triple #  7: 9*9 + 40*40 = 41*41 = 1681
Triple #  8: 28*28 + 45*45 = 53*53 = 2809
Triple #  9: 11*11 + 60*60 = 61*61 = 3721
Triple # 10: 16*16 + 63*63 = 65*65 = 4225
Triple # 11: 33*33 + 56*56 = 65*65 = 4225
Triple # 12: 48*48 + 55*55 = 73*73 = 5329
Triple # 13: 13*13 + 84*84 = 85*85 = 7225
Triple # 14: 36*36 + 77*77 = 85*85 = 7225
Triple # 15: 39*39 + 80*80 = 89*89 = 7921
Triple # 16: 65*65 + 72*72 = 97*97 = 9409
Triple # 17: 20*20 + 99*99 = 101*101 = 10201
.
.
```

ΑΣΚΗΣΗ-9

Δύο πρώτοι αριθμοί λέγονται δίδυμοι αν διαφέρουν κατά 2 (π.χ. 3 και 5, 101 και 103). Να γραφεί ένα πρόγραμμα που θα βρίσκει και θα εμφανίζει όλους τους θετικούς δίδυμους πρώτους αριθμούς που είναι μικρότεροι του 10000.

ΥΠΟΔΕΙΞΗ-1: για να ελέγξετε το πρόγραμμά σας : οι δύο τελευταίοι δίδυμοι που είναι μικρότεροι από 10000 είναι οι αριθμοί 9929 και 9931

ΥΠΟΔΕΙΞΗ-2 :

Ο παρακάτω κώδικας προγράμματος βρίσκει και εμφανίζει τους θετικούς πρώτους αριθμούς που υπάρχουν μέχρι ένα δεδομένο όριο n (n = γνωστό).

```
#include <iostream>
using namespace std;
void main()
{
      unsigned posprime, posDiv,n;
      do    {
                  cout<<"limit=? ( > 0 please) ";
                  cin>>n;
            }     while (n <= 0);
      cout << "Primes <= "<< n<< endl;
      for ( posprime = 2;posprime <= n;posprime++ )
      {     for (posDiv = 2;posDiv < posprime; posDiv++)
            if (0 == posprime%posDiv) break;
                  if (posDiv == posprime)
                        cout<<posprime<<endl;   }
}
```

ΑΣΚΗΣΗ-10

Να γραφεί πρόγραμμα σε γλώσσα C++ που θα εισάγει από το πληκτρολόγιο n (n = γνωστό) θετικούς αριθμούς τύπου float, με τους απαραίτητους ελέγχους εγκυρότητας τιμών. (Εναλλακτικά μπορεί να χρησιμοποιηθεί συνάρτηση δημιουργίας τυχαίων αριθμών). Μετά την εισαγωγή όλων των αριθμών το πρόγραμμα θα εμφανίζει μια αναφορά στατιστικών μεγεθών για τους αριθμούς αυτούς με την κατάλληλη μορφοποίηση.

Θα πρέπει να χρησιμοποιηθεί ο παρακάτω τύπος για τη μεταβλητότητα (variance) :

$$\frac{\sum_{i=1}^{n} x_i^2}{n} - \overline{X}^2$$

Όπου $\overline{X}$ είναι η μέση τιμή (mean). Η τυπική απόκλιση (standard deviation) είναι η τετραγωνική ρίζα της μεταβλητότητας.

Παράδειγμα

Αν εισαχθούν οι τιμές

```
3.1   77.6  19.2  54.9  18.6
```

η έξοδος θα είναι :

```
Statistical summary:
5 numbers read
      Maximum:          77.6
      Minimum:          3.1
      Sum:              173.4
      Mean:       34.68
      Variance:         749.293
      StdDev:           27.3732
```

ΑΣΚΗΣΗ-11

Για τη δημιουργία τυχαίων αριθμών χρησιμοποιείται η συνάρτηση **rand( )** που ανήκει στη cstdlib και δημιουργεί θετικούς ακέραιους αριθμούς (0-32767). Για τη χρήση της απαιτείται η δήλωση #include <cstdlib>. Για την παραγωγή τυχαίων αριθμών σε μια συγκεκριμένη περιοχή τιμών χρησιμοποιείται ο τελεστής % (modulus) π.χ. rand( ) % 10 → τιμές από 0 έως 9.

Η συνάρτηση rand στην πραγματικότητα δημιουργεί ψευδοτυχαίους αριθμούς, εκτός και αν επιλεγεί διαφορετικό σημείο εκκίνησης της γεννήτριας, μέσω τη συνάρτησης **srand** και του ορίσματος **seed**. Συνήθως ως seed χρησιμοποιείται το ρολόι του συστήματος μέσω της συνάρτησης time(0) που επιστρέφει τον αριθμό των δευτερολέπτων από το ρολόι του συστήματος. Έτσι χρησιμοποιείται η εντολή **srand(time(0))**. Για τη χρήση της time(0) απαιτείται η δήλωση #include <ctime>.

Να γραφεί μια συνάρτηση με όνομα random_gen που θα δημιουργεί τυχαίους ακέραιους θετικούς αριθμούς, σε πλήθος n (n>0). Οι αριθμοί πρέπει να βρίσκονται εντός των ορίων [α,β] (α<β). Τα όρια α,β και το πλήθος n θα δίνονται ως παράμετροι κατά τη κλήση της συνάρτησης random_gen από τη συνάρτηση main( ).

Στη συνέχεια να γραφεί ένα πρόγραμμα που θα καλεί τη συνάρτηση και θα εμφανίζει τους τυχαίους αριθμούς που δημιουργούνται. Οι πραγματικές τιμές των παραμέτρων α, β θα εισάγονται με την εντολή cin ενώ η τιμή n θα καθορίζεται μέσω της #define.

ΑΣΚΗΣΗ-12

Να γραφεί ένα πρόγραμμα που θα ορίζει τους πίνακες :

```
int i1[5], i2[5], sum[6]
```

Το πρόγραμμα θα περιμένει από το χρήστη να εισάγει σε καθένα από τους πίνακες i1 και i2 5 ψηφία. Κάθε ένας από τους πίνακες i1 και i2 θα αντιμετωπίζεται ως ένας ακέραιος αριθμός. Αν π.χ. στον πίνακα i1 εισαχθούν τα ψηφία 2, 7, 1, 5, 9 το πρόγραμμα θα αντιμετωπίζει τον πίνακα i1 ως τον ακέραιο 27159. Στη συνέχεια το πρόγραμμα θα προσθέτει τους πενταψήφιους ακέραιους στους πίνακες i1 και i2, θα αποθηκεύει το άθροισμα στον πίνακα sum και θα εμφανίζει στην οθόνη και τους τρεις πίνακες. Αν π.χ. οι τιμές των i1 και i2 είναι αντίστοιχα 27159 και 77118 τότε η μορφή των πινάκων i1 , i2 και sum θα είναι

2	7	1	5	9

7	7	1	1	8

1	0	4	2	7	7

ΑΣΚΗΣΗ-13

Ένας μονοδιάστατος πίνακας 2*N θέσεων (N = γνωστό και N>0) περιέχει, ανά ζεύγη, τις τιμές μετρήσεων ενός φυσικού μεγέθους και την αντίστοιχη συχνότητα εμφάνισης. Να γίνει εισαγωγή τιμών στον πίνακα (έστω x το όνομα του πίνακα) έτσι ώστε:

- Στις άρτιες θέσεις του πίνακα αντιστοιχούν οι τιμές των μετρήσεων του φυσικού μεγέθους. Οι τιμές που πρέπει να ανήκουν στην περιοχή [1-99].
- Στις περιττές θέσεις του πίνακα οι τιμές αντιστοιχούν στη συχνότητα εμφάνισης και πρέπει να ανήκουν στην περιοχή [1-9].

Με τον τρόπο αυτό δημιουργούνται διαδοχικά ζεύγη τιμών (*α, β*) για τα οποία θα ισχύει: 0 < *α* < 100 και 0 < *β* < 10. Η εισαγωγή των αριθμών μπορεί να γίνει με οποιονδήποτε τρόπο (εντολή scanf, χρήση συνάρτησης δημιουργίας τυχαίων αριθμών), αρκεί να υπάρχει έλεγχος εγκυρότητας τιμών.

Να θεωρήσετε ότι δίνεται επίσης ένας μονοδιάστατος πίνακας (έστω y το όνομα του πίνακα), M θέσεων (M = γνωστό και 0<M<10).

Να δημιουργήσετε ένα νέο πίνακα (έστω z το όνομα του πίνακα) που θα περιλαμβάνει τα στοιχεία α του αρχικού πίνακα x στα οποία αντιστοιχεί τιμή *β* ίση με κάποιο από τα στοιχεία του πίνακα y. Στη συνέχεια να εμφανίσετε τα στοιχεία του νέου πίνακα z.

Παράδειγμα:

Πίνακας x με N = 8 →

94	7	71	2	27	4	65	9	33	1	18	7	98	3	44	7

Πίνακας y με M=4 →

1	7	8	4

Πίνακας z →

33	94	18	44	27

ΑΣΚΗΣΗ - 14

Η παραγωγή ενός προϊόντος απαιτεί N φάσεις κατεργασίας από M σε πλήθος διαφορετικές εργαλειομηχανές (N = γνωστό , M = γνωστό). Κάθε εργαλειομηχανή μπορεί να χρησιμοποιηθεί περισσότερες από μία φορές. Για κάθε φάση κατεργασίας είναι γνωστός ο κωδικός της (ακέραιος αριθμός στην περιοχή [1-999]) και ο κωδικός της εργαλειομηχανής που χρησιμοποιείται (ακέραιος αριθμός στην περιοχή [1-9]).

Να γραφεί πρόγραμμα σε γλώσσα C που θα υλοποιεί τα εξής:

1. Θα καταχωρεί δεδομένα σε έναν πίνακα ακεραίων με 2*N σε πλήθος θέσεις, σύμφωνα με τα παραπάνω. Στις άρτιες θέσεις του πίνακα θα καταχωρούνται οι κωδικοί των φάσεων κατεργασίας και στις περιττές θέσεις του πίνακα οι κωδικοί των εργαλειομηχανών. (ΠΡΟΣΟΧΗ!! Εάν χρησιμοποιηθεί η εντολή scanf είναι απαραίτητος ο έλεγχος εγκυρότητας τιμών. Συνιστάται η κατάλληλη χρήση της συνάρτησης rand( ) για να αποφευχθεί η διαδικασία ελέγχου εγκυρότητας τιμών).
2. Θεωρώντας ότι η σειρά εκτέλεσης των φάσεων κατεργασίας σε κάθε εργαλειομηχανή καθορίζεται από τον κωδικό της φάσης κατεργασίας, με προτεραιότητα στις φάσεις με μικρότερο κωδικό φάσης κατεργασίας:

   Να βρείτε και να εμφανίσετε τη σειρά εκτέλεσης των φάσεων κατεργασίας σε κάθε εργαλειομηχανή, ΧΩΡΙΣ ΝΑ ΧΡΗΣΙΜΟΠΟΙΗΣΕΤΕ ΔΙΑΔΙΚΑΣΙΕΣ ΤΑΞΙΝΟΜΗΣΗΣ!

Αριθμητικό ΠΑΡΑΔΕΙΓΜΑ , N=10

145	2	180	5	19	8	91	2	577	8	91	2	72	5	227	8	33	8	759	5

Οι χρησιμοποιούμενες εργαλειομηχανές (M = 3) είναι οι : 2, 5, 8. (ΥΠΟΔΕΙΞΗ : οι χρησιμοποιούμενες εργαλειομηχανές πρέπει να βρεθούν από τον ανωτέρω πίνακα).

Αποτελέσματα :

ΕΡΓΑΛΕΙΟΜΗΧΑΝΗ : 2 91, 91, 145
ΕΡΓΑΛΕΙΟΜΗΧΑΝΗ : 5 72, 180, 759
ΕΡΓΑΛΕΙΟΜΗΧΑΝΗ : 8 19, 33, 227, 577

ΑΣΚΗΣΗ - 15

Οι *N* φάσεις κατεργασίας (*N* = γνωστό) που εκτελούνται στις εργαλειομηχανές μιας παραγωγικής μονάδας για την παραγωγή ενός προϊόντος εφοδιάζονται με :

i. τον αριθμό ταυτοποίησης της φάσης κατεργασίας (*id*, ακέραιος αριθμός από 1 – 99)
ii. τον κωδικό της εργαλειομηχανής (*machine*, ακέραιος αριθμός από 1 – 9)
iii. την κατάσταση (*status*) της εργαλειομηχανής με τις εξής επιτρεπτές τιμές:
    1 (σημαίνει ότι η εργαλειομηχανή είναι έτοιμη προς χρήση),
    2 (σημαίνει ότι η εργαλειομηχανή δεν είναι διαθέσιμη προς χρήση)

Ένας πίνακας a[3*N] περιέχει ζεύγη τιμών (*id, machine, status*). Να γραφεί μια συνάρτηση με όνομα *task_info* και ορίσματα εισόδου:

α) ένα μονοδιάστατο πίνακα ακεραίων θετικών αριθμών 3*N θέσεων, όπως ο πίνακας a.

β) έναν ακέραιο αριθμό *code* που αντιστοιχεί στον κωδικό της εργαλειομηχανής ($1 \le code \le 9$, απαιτείται ο σχετικός έλεγχος)

Η συνάρτηση *task_info* θα επιστρέφει στη συνάρτηση main( ) δύο νέους πίνακες :

- Ο ένας, έστω b, θα περιλαμβάνει όλες τις φάσεις κατεργασίας που μπορούν να εκτελεστούν στην συγκεκριμένη εργαλειομηχανή (δηλ. αυτήν με κωδικό *code*), δηλ. όλες τις φάσεις με *status*=1.
- Ο άλλος, έστω c, θα περιλαμβάνει όλες τις φάσεις κατεργασίας που δεν μπορούν να εκτελεστούν στη συγκεκριμένη εργαλειομηχανή (δηλ. αυτήν με κωδικό *code*), δηλ. όλες τις φάσεις με *status*=2.

Κάθε ένας από τους δύο αυτούς πίνακες θα πρέπει να περιέχει τους αριθμούς ταυτοποίησης (δηλ. τις φάσεις κατεργασίας *id*) σε αύξουσα σειρά του *id* , ΧΩΡΙΣ να χρησιμοποιηθεί διαδικασία ταξινόμησης.
(ΠΡΟΣΟΧΗ: η συνάρτηση ΔΕΝ θα πρέπει να περιέχει εντολές printf)

Στη συνέχεια να γραφεί ένα πρόγραμμα σε γλώσσα C που:

4. θα γεμίζει τον πίνακα a, με χρήση της συνάρτησης δημιουργίας τυχαίων αριθμών rand( ). Στην περίπτωση που μια φάση κατεργασίας προκύψει στα δεδομένα περισσότερες από μία φορές δεν δημιουργείται πρόβλημα (η φάση αυτή εκτελείται περισσότερες από μία φορές), αλλά θα εμφανίζεται μόνον μία φορά στους πίνακες αποτελεσμάτων.
5. θα καλεί τη συνάρτηση *task_info* και θα εμφανίζει τα αποτελέσματα που θα επιστρέφει η συνάρτηση.

ΠΑΡΑΔΕΙΓΜΑ με *N*=9 και *code*=7

Πίνακας a (δηλ. 9 τριάδες) - όσες θέσεις αφορούν *code*=7 είναι σε γκρίζο φόντο:

79	2	1	11	**7**	2	67	**7**	1	88	4	1	5	**7**	2	19	3	1	90	**7**	2	27	**7**	1	59	3	2

Αποτελέσματα :

27	67	← αντιστοιχεί σε status = 1

5	11	90	← αντιστοιχεί σε status = 2

ΑΣΚΗΣΗ-16

Να γραφεί μια αναδρομική συνάρτηση για τον υπολογισμό του τετραγώνου sq_n , ενός μη αρνητικού ακεραίου αριθμού n, χρησιμοποιώντας τις παρακάτω σχέσεις :

Για $n = 0$: $sq_0 = 0 \quad , d_0 = 1$

Για $n > 0$: $sq_n = sq_{n-1} + d_{n-1} \quad , d_n = d_{n-1} + 2$

Η τιμή d_n αντιπροσωπεύει πάντοτε τη διαφορά μεταξύ του τετραγώνου του αριθμού n και του τετραγώνου του αριθμού $n-1$. Για κάθε τιμή n η τιμή d_n αυξάνεται κατά 2. Οι τρεις πρώτες τιμές για το sq_n είναι 0, 1 και 4. Οι τρεις πρώτες τιμές για το d_n είναι 1,3 και 5.

Στη συνέχεια να γράψετε ένα πρόγραμμα (συνάρτηση main( )) που θα καλεί την αναδρομική συνάρτηση για τον υπολογισμό του τετραγώνου ενός μη αρνητικού ακέραιου αριθμού που θα εισάγεται μέσω της εντολής *cin*.

(ΥΠΟΔΕΙΞΗ : να εξετάσετε τον τρόπο δημιουργίας των τετραγώνων των αριθμών 1, 2, 3, 4 σύμφωνα με τις παραπάνω σχέσεις πριν προχωρήσετε στην κατασκευή της συνάρτησης).

ΑΣΚΗΣΗ-17

Να γραφούν συναρτήσεις για τον υπολογισμό του εμβαδού και του όγκου των εξής γεωμετρικών σχημάτων:

- ΕΜΒΑΔΟΝ (όνομα συνάρτησης area)
  - Τετράγωνο
  - Κύκλος
  - Τρίγωνο

- ΟΓΚΟΣ (όνομα συνάρτησης volume)
  - Ορθογώνιο παραλληλεπίπεδο
  - Σφαίρα
  - Πυραμίδα

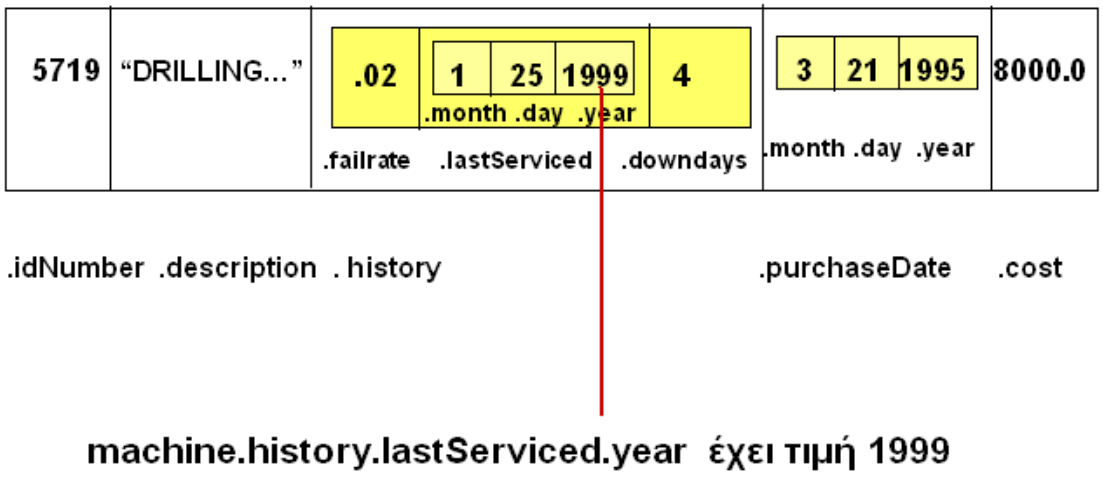
Στη συνέχεια να γραφεί ένα πρόγραμμα που θα δέχεται τα κατάλληλα δεδομένα για κάθε περίπτωση (μέσω της cin) και θα εμφανίζει τα αντίστοιχα αποτελέσματα στη main( ). **Να χρησιμοποιηθεί η υπερφόρτωση συναρτήσεων.**

ΑΣΚΗΣΗ - 18

Οι πληροφορίες για τις μηχανές που διαθέτει μια μονάδα παραγωγής περιλαμβάνουν :

1. αριθμό μηχανής (idnumber - τύπου int)
2. περιγραφή της μηχανής (description - string 80 χαρακτήρων)
3. ημερομηνία αγοράς (purchaseDate)
4. κόστος (cost - τύπου float)
5. το ιστορικό της (history)
    a. ποσοστό αποτυχίας (failrate - τύπου float)
    b. αριθμός ημερών που μένει εκτός λειτουργίας (downdays - τύπου int)
    c. ημερομηνία τελευταίας επισκευής (lastServiced)

Η μορφή μιας τυπικής δομής για μια μηχανή φαίνεται στο παρακάτω σχήμα :



Να γραφεί ένα πρόγραμμα σε γλώσσα C++ που θα υλοποιεί τα παρακάτω :

1. Θα ορίζει τις απαραίτητες δομές (structs) :
    - DateType (για τις ημερομηνίες με τη μορφή που φαίνονται στο παράδειγμα)
    - StatisticsType (θα περιλαμβάνει τα πεδία που αναφέρονται στο ιστορικό)
    - MachineRec (θα περιλαμβάνει όλες τις πληροφορίες που αφορούν τη μηχανή και με τη σειρά που φαίνονται στο παράδειγμα)
2. Θα δημιουργεί στο κυρίως πρόγραμμα ένα πίνακα δομών με 10 στοιχεία του τύπου MachineRec
3. χρησιμοποιώντας συναρτήσεις (functions) οι οποίες θα καλούνται κατάλληλα από τη main( ) :
    - Θα εισάγει από το πληκτρολόγιο δεδομένα στον πίνακα
    - Θα εμφανίζει τα στοιχεία του πίνακα
    - Θα υπολογίζει και θα εμφανίζει :
        - τη μέση τιμή κόστους των μηχανών
        - τη μέση τιμή των ημερών που οι μηχανές είναι εκτός λειτουργίας
        - το πλήθος των μηχανών που αγοράσθηκαν το 2007

ΑΣΚΗΣΗ-19

Το Καρτεσιανό και το Σφαιρικό Σύστημα Συντεταγμένων καθορίζουν τη θέση ενός σημείου του τρισδιάστατου επίπεδου χώρου μέσω των μεγεθών (x,y,z) και (r,ϑ,φ) αντίστοιχα. Ο μετασχηματισμός από τις Καρτεσιανές Συντεταγμένες (x,y,z) στις Σφαιρικές (r,ϑ,φ) πραγματοποιείται μέσω των τύπων:

$$r=\sqrt{x^2+y^2+z^2}, \quad \theta=\arctan\left(\sqrt{x^2+y^2},z\right), \qquad \varphi=\arctan(y,x)$$

Να ορίσετε μια δομή με όνομα **coords** για τη διαχείριση υλικών σημείων μάζας m_i ενός στερεού σώματος. Τα μέλη της δομής είναι η μάζα και οι τρεις καρτεσιανές συντεταγμένες (x,y,z), όπως ορίστηκαν προηγουμένως. Όλες οι τιμές είναι τύπου double.

Ένα σειριακό αρχείο με όνομα **xyz.txt** περιέχει άγνωστο πλήθος γραμμών που αντιστοιχούν στα υλικά σημεία ενός στερεού σώματος. Κάθε γραμμή του αρχείου περιέχει τέσσερις τιμές τύπου double που αντιστοιχούν στις τρεις καρτεσιανές συντεταγμένες και στη μάζα ενός υλικού σημείου (μέλη - δεδομένα).

Να οριστεί ένας πίνακας δομών τύπου **coords,** Ν θέσεων (Ν = γνωστό) .

Τα δεδομένα στον πίνακα δομών θα εισάγονται μέσω προσπέλασης του σειριακού αρχείου **xyz.txt** εφόσον το μέγεθος r που προκύπτει από τον μετασχηματισμό ικανοποιεί τη σχέση : $1.0 \le r \le 15.0$.

Αν το πλήθος των εγγραφών του αρχείου είναι μεγαλύτερο από την τιμή Ν τότε οι επιπλέον εγγραφές του αρχείου δεν θα συμπεριληφθούν στον πίνακα δομών.

Ζητούνται τα παρακάτω (συνάρτηση main( )):

1. να γίνει η εισαγωγή των δεδομένων από το αρχείο **xyz.txt** στον πίνακα δομών σύμφωνα με τα ανωτέρω, ελέγχοντας την ύπαρξη του αρχείου (δείτε τον κώδικα στη συνέχεια). Αν το αρχείο περιλαμβάνει περισσότερα από Ν υλικά σημεία να βρείτε και να εμφανίσετε το πλήθος των γραμμών του αρχείου που ικανοποιούν τη συνθήκη $1.0 \le r \le 15.0$ και δεν θα συμπεριληφθούν στον πίνακα δομών. Αν το αρχείο περιλαμβάνει λιγότερες από Ν γραμμές να βρείτε και να εμφανίσετε το πλήθος των γραμμών του αρχείου που θα περιέχει ο πίνακας δομών.
2. Στη συνέχεια να βρείτε και να εμφανίσετε το μέτρο του διανύσματος θέσης του κέντρου μάζας του στερεού σώματος από τον παρακάτω τύπο, λαμβάνοντας υπόψη όλα τα στοιχεία του πίνακα δομών.

$$\vec{r}_C=\frac{\sum_{i=1}^{N} m_i\vec{r}_i}{\sum_{i=1}^{N} m_i}$$

```
FILE *fp;
errno_t err;
…
      if (err=(fopen_s(&fp,filename,mode))!=0)
      {
            printf("error reading file....\n");
            exit(1);
      }
      else  // reading from file
      {
```

ΑΣΚΗΣΗ-20

Ένα σειριακό αρχείο με όνομα **points_2D.txt** περιέχει πληροφορίες που αντιστοιχούν σε χώρους με κινητές συσκευές. Σε κάθε γραμμή του αρχείου, που αφορά ένα χώρο, υπάρχουν δύο αριθμοί τύπου double που αντιστοιχούν στις συντεταγμένες του, και μια ακέραια τιμή που δηλώνει το πλήθος των κινητών συσκευών που υπάρχουν στο χώρο π.χ.

```
5.27    7.92    19
3.75    8.57    25
……..
```

Να ορίσετε μια δομή με όνομα **cell** για τη διαχείριση των πληροφοριών του αρχείου **points_2D.txt**. Τα μέλη της δομής είναι οι 2 καρτεσιανές συντεταγμένες (x,y) και το πλήθος των κινητών συσκευών m, όπως ορίστηκαν προηγουμένως.

Να οριστεί ένας πίνακας δομών τύπου **cell**, Ν θέσεων (Ν = γνωστό) .

Η εξυπηρέτηση των κινητών συσκευών θα γίνεται μέσω σταθμών βάσης (κεραίες). Έχουν προεπιλεγεί 2 σημεία, έστω Α και Β, για να αποτελέσουν σταθμούς βάσης που θα μπορούν να εξυπηρετούν ταυτόχρονα τις κινητές συσκευές. Για τα δύο αυτά σημεία είναι γνωστές οι επιφανειακές συντεταγμένες τους x_A, y_A, x_B, y_B που δίνονται ως σταθερές στην αρχή του προγράμματος. Λόγω εξασθένησης του σήματος από τον σταθμό βάσης μέχρι την κινητή συσκευή κάθε σταθμός βάσης μπορεί να καλύψει κινητές συσκευές μέχρι μια μέγιστη απόσταση από αυτόν.

Τα δεδομένα στον πίνακα δομών θα εισάγονται μέσω προσπέλασης του σειριακού αρχείου **points_2D.txt** εφόσον οι αποστάσεις του κάθε σημείου i του αρχείου και από τα δύο σημεία Α και Β, έστω $d_{A,i}$, $d_{B,i}$ ικανοποιούν συγχρόνως τις σχέσεις : $d_{A,i} \le 10.0$, $d_{B,i} \le 10.0$. Αν το πλήθος των εγγραφών του αρχείου είναι μεγαλύτερο από την τιμή Ν τότε οι επιπλέον εγγραφές του αρχείου δεν θα συμπεριληφθούν στον πίνακα δομών. Η απόσταση δύο σημείων (x_1, y_1) και (x_2, y_2) στο δισδιάστατο χώρο δίνεται από τη σχέση:

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} .$$

Ζητούνται τα παρακάτω (συνάρτηση main( )):

3. να γίνει η εισαγωγή των δεδομένων από το αρχείο **points_2D.txt** στον πίνακα δομών σύμφωνα με τα ανωτέρω, ελέγχοντας την ύπαρξη του αρχείου (δείτε τον κώδικα στη συνέχεια). Αν το αρχείο περιλαμβάνει περισσότερα από Ν σημεία να βρείτε και να εμφανίσετε το πλήθος των γραμμών του αρχείου που ικανοποιούν τις σχέσεις : $d_{A,i} \le 10.0$, $d_{B,i} \le 10.0$ και δεν θα συμπεριληφθούν στον πίνακα δομών. Αν το αρχείο περιλαμβάνει λιγότερες από Ν γραμμές να βρείτε και να εμφανίσετε το πλήθος των γραμμών του αρχείου που θα περιέχει ο πίνακας δομών.
4. Στη συνέχεια να βρείτε και να εμφανίσετε :
   a. Το συνολικό πλήθος των κινητών συσκευών που θα μπορούν να εξυπηρετηθούν από τα σημεία Α και Β χρησιμοποιώντας τα στοιχεία του πίνακα δομών.
   b. Τη μέση τιμή των αποστάσεων όλων των σημείων του πίνακα δομών από το σημείο Α και από το σημείο Β (δηλ. 2 μέσες τιμές).

```
FILE *fp;
errno_t err;
…
      if (err=(fopen_s(&fp,filename,mode))!=0)
      {     printf("error reading file....\n");
            exit(1);
      }
      else  {             //reading from file
```

ΑΣΚΗΣΗ-21

Το πρόβλημα της εύρεσης του πλησιέστερου ζεύγους σημείων είναι ένα γεωμετρικό πρόβλημα που αφορά την εύρεση δύο σημείων στο χώρο που βρίσκονται πλησιέστερα το ένα στο άλλο. Η απόσταση δύο σημείων (x_1, y_1, z_1) και (x_2, y_2, z_2) είναι :

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

Ένα σειριακό αρχείο με όνομα **points3D.txt** περιέχει σε κάθε γραμμή του τις καρτεσιανές συντεταγμένες σημείων στον τρισδιάστατο χώρο:
π.χ.

18.7 17.9 5.26
10.55 5.37 2.15
110.22 77.44 27.19

Να ορίσετε μια δομή με όνομα **point_3D** για τη διαχείριση των συντεταγμένων των σημείων αυτών. Τα μέλη της δομής είναι οι τρεις προαναφερόμενοι αριθμοί, τύπου double.
Να οριστεί ένας πίνακας δομών τύπου **point_3D**, Ν θέσεων (Ν = γνωστό) .
Τα δεδομένα στον πίνακα δομών θα εισάγονται μέσω προσπέλασης του σειριακού αρχείου **points3D.txt**.

Να θεωρήσετε ως δεδομένο ότι η τιμή του Ν υπερκαλύπτει το πλήθος των εγγραφών του αρχείου.

Ζητούνται τα παρακάτω (συνάρτηση main( )):

1. να γίνει η εισαγωγή των δεδομένων από το αρχείο **points3D.txt** στον πίνακα δομών σύμφωνα με τα ανωτέρω, ελέγχοντας την ύπαρξη του αρχείου (δείτε τον κώδικα στη συνέχεια) και να βρείτε και να εμφανίσετε το πλήθος των γραμμών του αρχείου που θα περιέχει ο πίνακας δομών.
2. Χρησιμοποιώντας όλα τα στοιχεία του πίνακα δομών να βρείτε και να εμφανίσετε όλα τα ζεύγη σημείων με την ίδια ελάχιστη απόσταση μεταξύ τους, με ακρίβεια υπολογισμού 10^{-3}. Στο τέλος να εμφανίσετε και την τιμή αυτής της ελάχιστης απόστασης.

```
FILE *fp;
errno_t err;
…
      if (err=(fopen_s(&fp,filename,mode))!=0)
      {
            printf("error reading file....\n");
            exit(1);
      }
      else  // reading from file
      {
```

ΑΣΚΗΣΗ-22

Για οποιονδήποτε θετικό ακέραιο αριθμό n , ο κύβος του, n^3 , είναι το άθροισμα όλων των περιττών αριθμών που περιλαμβάνονται στο διάστημα $\left[n^2 - n + 1,\ n^2 + n - 1\right]$.

Παράδειγμα : Για $n = 6,\quad n^2 - n + 1 = 31$ και $31 + 33 + 35 + 37 + 39 + 41 = 216 \quad [= n^2 + n - 1] \quad = 6^3$.

Να γραφεί πρόγραμμα σε γλώσσα C++ που θα επαληθεύει τη σχέση αυτή για κάθε μονοψήφιο και διψήφιο ακέραιο θετικό αριθμό.

ΑΣΚΗΣΗ-23

Μια εργαλειομηχανή λειτουργεί με 2 περιστρεφόμενους άξονες κυκλικής διατομής με γνωστές διαμέτρους $D1, D2$ αντίστοιχα. Λόγω της λειτουργίας των αξόνων προκύπτουν αλλοιώσεις της γεωμετρίας τους σε διάφορα σημεία, με αποτέλεσμα οι τομές τους σε διάφορα σημεία κατά το μήκος τους να μην είναι πλήρως κυκλικές.

Ένα σειριακό αρχείο με όνομα **crs.txt** περιέχει σε κάθε γραμμή του :

- 2 αριθμούς τύπου double που αντιστοιχούν στο μήκος, σε mm, δύο κάθετων μεταξύ τους διαμέτρων μιας τομής κάθετης στον άξονα περιστροφής κάθε άξονα.
- Έναν αριθμό τύπου double που αντιστοιχεί στην απόσταση της τομής, σε mm, από την αρχή του άξονα
- Έναν ακέραιο θετικό αριθμό, 1 ή 2 , που δηλώνει τον άξονα περιστροφής

π.χ. για $D1 = 110, D2 = 180$

```
180.81  179.93  0.22    2
109.55  109.57  0.15    1
110.22  110.44  0.19    1
…
```

Να ορίσετε μια δομή με όνομα **crd** για τη διαχείριση των διαστάσεων των διαμέτρων των τομών αυτών. Τα private μέλη-δεδομένα της δομής είναι οι τέσσερεις προαναφερόμενοι αριθμοί.

Να οριστεί ένας πίνακας δομών τύπου **crd,** Ν θέσεων (Ν = γνωστό) .

Τα δεδομένα στον πίνακα δομών θα εισάγονται μέσω προσπέλασης του σειριακού αρχείου **crs.txt** εφόσον η απόλυτη τιμή της διαφοράς των δύο διαμέτρων κάθε γραμμής του αρχείου είναι μεγαλύτερη από το 2% της διαμέτρου του αντίστοιχου άξονα ($D1$ ή $D2$) .

Να θεωρήσετε ως δεδομένο ότι η τιμή του Ν υπερκαλύπτει το πλήθος των εγγραφών του αρχείου.

Ζητούνται τα παρακάτω (συνάρτηση main( )):

3. να γίνει η εισαγωγή των δεδομένων από το αρχείο **crs.txt** στον πίνακα δομών σύμφωνα με τα ανωτέρω, ελέγχοντας την ύπαρξη του αρχείου (δείτε τον κώδικα στη συνέχεια) και να βρείτε και να εμφανίσετε το πλήθος των γραμμών του αρχείου που θα περιέχει ο πίνακας δομών.
4. Χρησιμοποιώντας όλα τα στοιχεία του πίνακα δομών να βρείτε και να εμφανίσετε :
   a. το πλήθος των τομών κάθε άξονα
   b. τη μέση τιμή των αποκλίσεων κάθε μιας από τις δύο μετρούμενες διαμέτρους κάθε άξονα από την αρχική διάμετρο κάθε άξονα, χωρίς να συμπεριληφθούν οι μηδενικές αποκλίσεις, με ακρίβεια 10^{-4}
      (ΥΠΟΔΕΙΞΗ : πρέπει να προσδιορίσετε 4 τιμές , 2 για κάθε άξονα)

```
FILE *fp;
errno_t err;
…
      if (err=(fopen_s(&fp,filename,mode))!=0)
      {
            printf("error reading file....\n");
            exit(1);
      }
      else  // reading from file
      {
```

ΑΣΚΗΣΗ-24

Να γραφεί αλγόριθμος ή πρόγραμμα σε γλώσσα C++ που χρησιμοποιώντας κατάλληλα το παρακάτω τμήμα κώδικα σε γλώσσα C: Θα εξετάζει όλους τους πρώτους (primes) αριθμούς που είναι μικρότεροι από το 5000 για να βρίσκει και να εμφανίζει τη συχνότητα των αποστάσεων κάθε ζεύγους διαδοχικών πρώτων αριθμών. Δηλαδή, αν η ακολουθία τω πρώτων αριθμών είναι :

2	3	5	7	11	13	17	19	23	29	31	37	41	43	47	53	…..	4993	4999

τότε οι διαφορές για κάθε διαδοχικό ζεύγος είναι:

1	2	2	4	2	4	2	4	6	2	6	4	….	6

και πρέπει να βρεθούν και να εμφανιστούν οι συχνότητες αυτών των διαφορών, δηλ. πόσα 1, 2, 4, 6 κλπ υπάρχουν.

(ΥΠΟΔΕΙΞΗ : Το παρακάτω τμήμα κώδικα σε γλώσσα C βρίσκει αν ένας ακέραιος και θετικός αριθμός $k \geq 2$ είναι πρώτος (prime) αριθμός).

```
i=2; flag=0;
while ((i<=k/2) && (flag==0))
{
      if (k%i==0) flag=1;
      i++;
}
if (flag==0) printf("number %4d  is prime \n",k);
```

ΑΣΚΗΣΗ-25

Ένας κυκλικός πρώτος (prime) αριθμός είναι εκείνος που παραμένει πρώτος (prime) μετά από μία επαναληπτική τοποθέτηση του πρώτου ψηφίου στο τέλος π.χ. ο αριθμός 197 είναι κυκλικός prime διότι οι αριθμοί 197 , 971 και 719 είναι πρώτοι (prime) αριθμοί.

Να γραφεί αλγόριθμος ή πρόγραμμα σε γλώσσα C++ που θα βρίσκει και θα εμφανίζει όλους αυτούς τους κυκλικούς διψήφιους και τριψήφιους πρώτους αριθμούς. Σε κάθε βήμα θα πρέπει να εμφανίζονται και οι δύο ή και οι τρεις προκύπτοντες πρώτοι αριθμοί.

(ΥΠΟΔΕΙΞΗ : Το παρακάτω τμήμα κώδικα σε γλώσσα C++ βρίσκει αν ένας ακέραιος και θετικός αριθμός $k \geq 2$ είναι πρώτος (prime) αριθμός).

```
i=2; flag=0;
while ((i<=k/2) && (flag==0))
{
      if (k%i==0) flag=1;
      i++;
}
if (flag==0) printf("number %4d  is prime \n",k);
```

ΑΣΚΗΣΗ-26

Πίνακας ακεραίων Ν θέσεων (Ν= γνωστό και Ν>300), έστω a, περιέχει ακέραιες θετικές τιμές στην περιοχή [1,9]. Να θεωρήσετε ότι οι τιμές αυτές αντιστοιχούν σε ορισμένα χρώματα. Ένας δεύτερος πίνακας, έστω b, με πλήθος θέσεων Μ=4 περιέχει ένα συγκεκριμένο μοτίβο τεσσάρων οποιωνδήποτε χρωμάτων από αυτά που υπάρχουν στον πίνακα a.

Να γραφεί πρόγραμμα σε γλώσσα C που θα βρίσκει και θα εμφανίζει:

Σε ποιες θέσεις του πίνακα a υπάρχει το μοτίβο που περιέχεται στον πίνακα b. Το πρόγραμμα θα πρέπει να εμφανίζει, για κάθε εμφάνιση του συγκεκριμένου μοτίβου, τη θέση έναρξης του μοτίβου και τη θέση πέρατος του μοτίβου

ΠΑΡΑΔΕΙΓΜΑ

Πίνακας a , με N = 21

2	6	7	8	1	4	9	4	3	7	8	1	4	5	5	7	1	8	3	2	9

Πίνακας b:

7	8	1	4

ΑΠΟΤΕΛΕΣΜΑΤΑ

1η εμφάνιση : έναρξη θέση 3, πέρας θέση 6

2η εμφάνιση : έναρξη θέση 10, πέρας θέση 13

ΑΣΚΗΣΗ-27

Αριθμητικός πίνακας ακεραίων, Ν θέσεων περιέχει όλους τους πρώτους (primes) ακέραιους θετικούς αριθμούς που είναι μικρότεροι του αριθμού 1000. Δίνεται ότι το πλήθος των αριθμών αυτών **δεν είναι** μεγαλύτερο από 200. Χρησιμοποιώντας τον κώδικα που δίνεται στη συνέχεια:

1. Να δημιουργήσετε αυτόν τον πίνακα και να καταχωρήσετε, σε αύξουσα διάταξη, τους πρώτους αριθμούς που ζητούνται (ΥΠΟΔΕΙΞΗ : Ο πρώτος prime αριθμός είναι το 2).
2. Να βρείτε και να εμφανίσετε το πλήθος αυτών των αριθμών.
3. Για κάθε δυνατό ζεύγος πρώτων αριθμών του πίνακα αυτού να βρείτε τη διαφορά τους και να την εμφανίσετε μαζί με τους δύο πρώτους αριθμούς, ανά γραμμή.
4. Να βρείτε και να εμφανίσετε, σε αύξουσα διάταξη, τη συχνότητα εμφάνισης όλων αυτών των διαφορών.

ΠΑΡΑΔΕΙΓΜΑ: Για ένα τμήμα του πίνακα

……	353	359	367	373	379	383	389	…….	997

Αριθμοί	ΔΙΑΦΟΡΑ
353, 359	6
359, 367	8
367, 373	6
373, 379	6
379, 383	4
383, 389	6

ΔΙΑΦΟΡΑ	4	6	8
Συχνότητα εμφάνισης	1	4	1

```
#include <stdio.h>

int n, i, c = 0;
void main()
{
printf("Enter any number n:");
scanf_s("%d", &n);

for (i = 1; i <= n; i++)
{ if (n % i == 0) c++;  }

if (c == 2) printf("%5d is a Prime number\n",n);
else     printf("%5d is not a Prime number\n",n);
}
```

ΑΣΚΗΣΗ-28

Ένας ακέραιος θετικός αριθμός μπορεί να γραφεί ως άθροισμα δύο διαφορετικών πρώτων (primes) αριθμών. Παράδειγμα για τον αριθμό 16 (ο πρώτος prime αριθμός είναι το 2) :

16 = 2 + 14	Το 2 είναι πρώτος, το 14 όχι
16 = 3 + 13	Και οι δύο είναι πρώτοι
16 = 4 + 12	Κανένας δεν είναι πρώτος
16 = 5 + 11	Και οι δύο είναι πρώτοι
16 = 6 + 10	Κανένας δεν είναι πρώτος
16 = 7 + 9	Το 7 είναι πρώτος, το 9 όχι
16 = 8 + 8	Οι αριθμοί είναι ΙΔΙΟΙ !!!!

1. Να γραφεί μία συνάρτηση με όνομα ***find_prime*** που θα δέχεται ως παράμετρο έναν ακέραιο θετικό αριθμό z και θα επιστρέφει εάν ο αριθμός αυτός είναι πρώτος (prime). Να χρησιμοποιήσετε τον πηγαίο κώδικα που δίνεται στο τέλος. (ΠΡΟΣΟΧΗ: η συνάρτηση ΔΕΝ θα πρέπει να περιέχει εντολές printf).

2. Να γραφεί μια συνάρτηση με όνομα ***primes_sum*** και παράμετρο εισόδου έναν ακέραιο θετικό αριθμό m . Η συνάρτηση θα επιστρέφει στη main( ) έναν πίνακα, με όνομα b , που θα περιέχει όλα τα ζεύγη πρώτων αριθμών, το άθροισμα των οποίων είναι ίσο με τον αριθμό m. Τα ζεύγη θα δημιουργούνται ταξινομημένα σε αύξουσα διάταξη ως προς το πρώτο στοιχείο κάθε ζεύγους. (ΠΡΟΣΟΧΗ: η συνάρτηση ΔΕΝ θα πρέπει να περιέχει εντολές printf).

3. Μονοδιάστατος αριθμητικός πίνακας ακεραίων, a , περιέχει N ακεραίους αριθμούς (N = γνωστό και $N > 20$) . Οι αριθμοί ανήκουν στο διάστημα [100 - 999]. Στη συνάρτηση main( ):
   a. Να καταχωρήσετε τιμές στα στοιχεία του πίνακα a εντός των αποδεκτών ορίων τιμών, με κατάλληλη χρήση της συνάρτησης δημιουργίας τυχαίων αριθμών rand().
   b. Για κάθε στοιχείο του πίνακα a να βρείτε και να εμφανίσετε όλα τα ζεύγη των πρώτων (primes) αριθμών το άθροισμα των οποίων ισούται με το στοιχείο αυτό, χρησιμοποιώντας τις συναρτήσεις ***find_prime*** και ***primes_sum.*** Σε κάθε γραμμή της οθόνης θα εμφανίζεται το στοιχείο του πίνακα και όλα τα σχετικά ζεύγη. Στο τέλος θα εμφανίζεται και το πλήθος αυτών των ζευγών.

ΑΡΙΘΜΗΤΙΚΟ ΠΑΡΑΔΕΙΓΜΑ

$m = 250$ ζεύγη : (11,239), (17, 233), (23,227), (53,197), (59,191), (71,179), (83,167), (101,149), (113,137)

```
#include <stdio.h>
void main()
{
    int n, i, flag = 0;

    printf("Enter a positive integer: ");
    scanf("%d",&n);

    for(i=2; i<=n/2; ++i)
    {
        // condition for nonprime number
        if(n%i==0) { flag=1; break; }
    }

    if (flag==0) printf("%d is a prime number.",n);
    else printf("%d is not a prime number.",n);
}
```

ΑΣΚΗΣΗ-29

Έστω n, k είναι δύο μη αρνητικοί ακέραιοι αριθμοί με $0 \le k \le n$. Ο διωνυμικός συντελεστής $\binom{n}{k}$ ορίζεται ως

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!} = \frac{n(n-1)(n-2)\cdots(n-k+1)}{1\cdot 2\cdot 3\cdots k}$$

Επίσης ορίζονται :

$$\binom{n}{0} = \binom{n}{n} = 1, \qquad \binom{n}{1} = \binom{n}{n-1} = n$$

Παράδειγμα :

$$\binom{12}{7} = \frac{12\cdot 11\cdot 10\cdot 9\cdot 8\cdot 7\cdot 6}{1\cdot 2\cdot 3\cdot 4\cdot 5\cdot 6\cdot 7} = 792$$

Να γραφεί πρόγραμμα σε γλώσσα C που θα υλοποιεί τα παρακάτω:

1. Θα εισάγει σε έναν μονοδιάστατο πίνακα ακεραίων θετικών αριθμών N (N = γνωστό) ζεύγη τιμών n, k. Οι τιμές των n, k θα πρέπει να ικανοποιούν και τους δύο ακόλουθους περιορισμούς:

   $$0 \le k \le n \qquad \textbf{και} \qquad 0 \le n \le 15$$

   Προτείνεται η εισαγωγή των τιμών να γίνεται με κατάλληλη χρήση της συνάρτησης rand().

2. Για κάθε ζεύγος τιμών του πίνακα αυτού:

   i. να καταχωρούνται σε ένα νέο μονοδιάστατο πίνακα, κατά σειρά, οι αριθμοί που συνθέτουν τον αριθμητή και τον παρονομαστή του τύπου του διωνυμικού συντελεστή, δηλ. για το ανωτέρω παράδειγμα ο νέος πίνακας θα είναι

12	11	10	9	8	7	6	1	2	3	4	5	6	7

   ii. Να εμφανίζονται τα περιεχόμενα του νέου πίνακα, μία σειρά για κάθε ζεύγος του αρχικού πίνακα

   iii. Να υπολογίζεται και να εμφανίζεται το αποτέλεσμα των πράξεων (δηλ. για το παράδειγμα η τιμή 792). Οι πράξεις θα πρέπει να γίνονται χρησιμοποιώντας ΜΟΝΟΝ τα στοιχεία του νέου πίνακα.

(Στις απαντήσεις σας πρέπει να ληφθούν υπόψη όλες οι δυνατές περιπτώσεις για τις τιμές των ζευγών n, k)

ΑΣΚΗΣΗ-30

Να δημιουργηθεί αριθμητικός πίνακας ακεραίων, μεγέθους *N* ζευγών (*N* = γνωστό). Κάθε ζεύγος τιμών (έστω a, b τα ονόματα των τιμών κάθε ζεύγους) περιέχει:

- Έναν τριψήφιο ακέραιο αριθμό [100,999], είναι ο a που αναφέρεται ανωτέρω
- Έναν μονοψήφιο ακέραιο αριθμό [1,9], είναι ο b που αναφέρεται ανωτέρω

Να γράψετε πρόγραμμα σε γλώσσα C που θα υλοποιεί τα παρακάτω:

1. Για κάθε ζεύγος τιμών (a, b) του πίνακα να βρείτε και να εμφανίσετε, όπως φαίνεται στο παράδειγμα που ακολουθεί, τους b σε πλήθος πρώτους (primes) αριθμούς που ικανοποιούν τη συνθήκη:

   $$a \leq prime\ number\ \leq 999.$$

   Να εξετάσετε και την περίπτωση να προκύπτουν λιγότεροι σε πλήθος πρώτοι αριθμοί λόγω αυτού του περιορισμού.

2. Επίσης, το πρόγραμμα θα βρίσκει και θα εμφανίζει, συμπεριλαμβάνοντας τα αποτελέσματα που προκύπτουν για όλα τα ζεύγη τιμών του πίνακα:
   - Το πλήθος των primes που βρέθηκαν (για το παράδειγμα το πλήθος είναι 17)
   - Το πλήθος των διαφορετικών primes που βρέθηκαν (για το παράδειγμα το πλήθος είναι 12)
   - Τη συχνότητα εμφάνισης κάθε prime, σε αύξουσα διάταξη των primes, **χωρίς να χρησιμοποιηθεί διαδικασία ταξινόμησης**

Για να διαπιστώσετε εάν ένας αριθμός είναι prime θα χρησιμοποιήσετε κατάλληλα και **ΜΟΝΟΝ** το παρακάτω πρόγραμμα που βρίσκει εάν ένας ακέραιος θετικός αριθμός, μεγαλύτερος ή και ίσος του 2, είναι πρώτος (prime).

ΠΑΡΑΔΕΙΓΜΑ: Αρχικός πίνακας για *N*=4 ζεύγη (η 1η γραμμή στον πίνακα του παραδείγματος, με τα ονόματα a και b, είναι για διευκόλυνσή σας)

a	b	a	b	a	b	a	b
945	**8**	**211**	**4**	**989**	**5**	**207**	**3**

ζεύγος	primes αριθμοί	Πλήθος primes
945, 8	**947, 953, 967, 971, 977, 983, 991, 997**	8
211, 4	**211, 223, 227, 229**	4
989, 5	**991, 997**	2
207,3	**211, 223, 227**	3

```
#include<stdio.h>
void main()
{
   int n, c = 2;
   printf("Enter a number to check if it is prime\n");
   scanf_s("%d",&n);
   for ( c = 2 ; c <= n - 1 ; c++ )
   {
      if ( n % c == 0 )
      { printf("%d isn't prime.\n", n);     break; }
   }
   if ( c == n ) printf("%d is prime.\n", n);

}
```

ΑΣΚΗΣΗ-31

Θεωρούμε έναν πίνακα ακεραίων θετικών αριθμών, έστω ***a***, μεγέθους ***N*** (***N*** = γνωστό), και έναν θετικό ακέραιο ***k***. Θέλουμε να βρούμε όλα τα ζεύγη στοιχείων του πίνακα ***a*** το άθροισμα των οποίων διαιρείται (ακριβώς) με το ***k***.
Παράδειγμα:
Έστω ***a***[ ]={ 2, 2, 1, 7, 5, 3}, ***N*** = 6, ***k*** = 4
Τα ζεύγη συνολικά είναι 5, τα εξής : (2, 2), (1, 7), (1, 3) , (7, 5), και (5, 3)

Ζητούνται τα εξής :

1. Να γραφεί μία συνάρτηση, σε γλώσσα προγραμματισμού C, με όνομα ***find_pairs*** και τις εξής παραμέτρους :

   ΕΙΣΟΔΟΥ
   - Ο ως άνω πίνακας ***a*** και ο θετικός ακέραιος ***k***

   ΕΞΟΔΟΥ
   - Ένας μονοδιάστατος αριθμητικός πίνακας, ***b***, που θα περιέχει τα ζεύγη που αναφέρονται ανωτέρω.
   - Ένας μονοδιάστατος αριθμητικός πίνακας, ***c***, που θα περιέχει τους primes (πρώτους) αριθμούς που υπάρχουν στον πίνακα ***b***, σε αύξουσα διάταξη, όσες φορές υπάρχει ο καθένας. Για την εύρεση των primes αριθμών : να γραφεί μία συνάρτηση με όνομα ***find_prime*** που θα δέχεται ως παράμετρο έναν ακέραιο θετικό αριθμό ***z*** και θα επιστρέφει εάν ο αριθμός αυτός είναι πρώτος (prime). Να χρησιμοποιήσετε τον πηγαίο κώδικα που δίνεται στο τέλος.

   (ΠΡΟΣΟΧΗ: η συνάρτηση ΔΕΝ θα πρέπει να περιέχει εντολές printf).

2. Στη συνάρτηση main( ):
   a. Να δημιουργηθεί ένας μονοδιάστατος πίνακας ακεραίων θετικών αριθμών, μεγέθους ***M*** (***M*** = γνωστό) και να καταχωρηθούν σε αυτόν τιμές, στο διάστημα [1,99] (αποκλειστικά με χρήση της rand( )).
   b. Να κληθεί η συνάρτηση ***find_pairs*** με ορίσματα εισόδου :
      i. τον πίνακα που δημιουργήθηκε στο ερώτημα 2α, και
      ii. τη μέση τιμή των στοιχείων του πίνακα του ερωτήματος 2α ως τιμή για το ***k***.
   c. Να εμφανίσετε τα αποτελέσματα που προκύπτουν από την κλήση της συνάρτησης ***find_pairs.***

```
#include <stdio.h>
void main()
{
    int n, i, flag = 1;

    printf("Enter a positive integer: ");
    scanf_s("%d",&n);

    for(i=2; i<=n/2; i++)
    {
        if(n%i==0) { flag=0; break; }
    }
    if (flag = = 1) printf("%d is a prime number.",n);
    else printf("%d is not a prime number.",n);
}
```

ΑΣΚΗΣΗ-32

Κάθε πρώτος (prime) αριθμός διαιρείται μόνον με τη μονάδα και με τον εαυτό του. Ο πρώτος prime αριθμός είναι το 2. Στη συνέχεια παρατίθεται μία περιοχή συνεχόμενων primes αριθμών, μικρότερων του 1000:

…. 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, ….

1. Να γραφεί μία συνάρτηση με όνομα ***find_prime*** που θα δέχεται ως παράμετρο έναν ακέραιο θετικό αριθμό z και θα επιστρέφει εάν ο αριθμός αυτός είναι πρώτος (prime). Να χρησιμοποιήσετε τον πηγαίο κώδικα που δίνεται στο τέλος. (ΠΡΟΣΟΧΗ: η συνάρτηση ΔΕΝ θα πρέπει να περιέχει εντολές printf).

2. Να γραφεί μια συνάρτηση με όνομα ***primes_triplets*** και παράμετρο εισόδου έναν πίνακα ακεραίων θετικών αριθμών, έστω a μεγέθους n. Η συνάρτηση θα επιστρέφει στη main( ) :

   a. Ένα νέο πίνακα, με όνομα b, που θα περιέχει σε αύξουσα σειρά διάταξης, όλα τα στοιχεία του πίνακα a, που είναι πρώτοι (primes) αριθμοί, χρησιμοποιώντας τη συνάρτηση ***find_prime*** (ΜΟΝΑΔΕΣ 2)

   b. Ένα νέο πίνακα, έστω c που θα περιέχει τριάδες ακεραίων αριθμών, ως εξής (ΠΡΟΣΟΧΗ : Η συνάρτηση δεν θα πρέπει να περιέχει εντολές printf):

      i. Ο πρώτος αριθμός σε κάθε τριάδα θα είναι κάθε αριθμός του πίνακα a που δεν είναι πρώτος (prime).

      ii. Ο δεύτερος αριθμός σε κάθε τριάδα θα είναι ο αμέσως μικρότερος prime αριθμός από τον πρώτο αριθμό της τριάδας.

      iii. Ο τρίτος αριθμός σε κάθε τριάδα θα είναι ο αμέσως μεγαλύτερος prime αριθμός από τον πρώτο αριθμό της τριάδας.

      ΠΑΡΑΔΕΙΓΜΑ: αν ο αριθμός από τον πίνακα a είναι ο 627 η τριάδα θα είναι 627, 619, 631

3. Μονοδιάστατος αριθμητικός πίνακας ακεραίων, x, περιέχει N ακεραίους αριθμούς (N = γνωστό και $N > 20$), στο διάστημα [100 - 999]. Στη συνάρτηση main( ):
   a. Να καταχωρήσετε τιμές στα στοιχεία του πίνακα x, με κατάλληλη χρήση της συνάρτησης δημιουργίας τυχαίων αριθμών rand().
   b. Να καλέσετε τη συνάρτηση ***primes_triplets***, με παράμετρο εισόδου τον πίνακα , και να εμφανίσετε τα αποτελέσματα που προκύπτουν από την κλήση της συνάρτησης.

```
#include <stdio.h>
void main()
{
    int n, i, flag = 1;

    printf("Enter a positive integer: ");
    scanf("%d",&n);

    for(i=2; i<=n/2; ++i)
    {
        if(n%i==0) { flag=0; break; }
    }
    if (flag = = 1) printf("%d is a prime number.",n);
    else printf("%d is not a prime number.",n);
}
```