ΕΡΩΤΗΜΑ 1^{Ο}

Έχουμε για την κάθε ρουτίνα ξεχωριστά:

1^{η} ρουτίνα

```
/*
 * C program to accept numbers as an input from user
 * and to sort them in ascending order.
 */
#include <stdio.h>

/* Fuction for getting sorting number in ascending order*/
void sort_numbers_ascending(int number[], int count)
{
   int temp, i, j, k;

   for (j = 0; j < count; ++j)
   {
      for (k = j + 1; k < count; ++k)
      {
         if (number[j] > number[k])
         {
            temp = number[j];
            number[j] = number[k];
            number[k] = temp;
         }
      }
   }
   printf("Numbers in ascending order:\n");
   for (i = 0; i < count; ++i)
      printf("%d\n", number[i]);
}
```

Τελεστές	Αριθμός εμφανίσεων	Έντελα	Αριθμός εμφανίσεων
void	1	count	2
int	3	i	4
number[]	7	j	7
,	5	k	7
sort_numbers _ascending()	1	temp	3
for(; ;)	3	"Numbers in ascending order:\n"	1
if()	1	"%d\n"	1
{}	4	0	2
printf()	2		
<	3		
>	1		
=	6		
;	6		
++	3		
+	1		
n1=15	N1=47	n2=8	N2=27

<u>Παραδοχές</u>

1. Παίρνουμε την συνάρτηση sort_numbers_ascending() μαζί με τις παρενθέσεις σαν έναν τελεστή στον ορισμό της, καθώς δεν γίνεται να οριστεί ορθά χωρίς αυτές.
2. Δεν συμπεριλαμβάνουμε τα σχόλια σαν έντελα ή τελεστές, καθώς δεν αποτελούν μέρος του κώδικα.
3. Δεν συμπεριλαμβάνουμε το include σαν τελεστή ή σαν έντελο, καθώς βρίσκεται εκτός της ρουτίνας.
4. Παίρνουμε τον τελεστή for(; ;) σαν έναν, καθώς ο τελεστής for στην c εμφανίζεται υποχρεωτικά με ζεύγος παρενθέσεων και 2 ερωτηματικά.

2η ρουτίνα

```
void main()
{
   int i, count, number[20], t=0;

   printf("How many numbers you are going to enter:");
   scanf("%d", &count);
   printf("\nEnter the numbers one by one:");

   while (t>20)
   {
    printf("\nThis is a test");
    scanf("%d", &count);
   }
   for (i = 0; i < count; ++i)
      scanf("%d", &number[i]);
 /* Calling the Function*/
   sort_numbers_ascending(number, count);
}
```

<u>Παραδοχές</u>

1.Παίρνουμε τις συναρτήσεις main(), sort_numbers_ascending() μαζί με τις παρενθέσεις σαν έναν τελεστή στον ορισμό και στην κλήση τους, καθώς δεν γίνεται να καλεστούν ορθά χωρίς αυτές.

2.Συμπεριλαμβάνουμε τα σχόλια σαν έντελα και τελεστές, καθώς αποτελούν μέρος του κώδικα.

3.Παίρνουμε τον τελεστή for(; ;) σαν έναν, καθώς ο τελεστής for στην c εμφανίζεται υποχρεωτικά με ζεύγος παρενθέσεων και 2 ερωτηματικά.

Τελεστές	Αριθμός εμφανίσεων	Έντελα	Αριθμός εμφανίσεω
Void	1	i	4
main()	1	count	5
Int	1	t	2
number[]	2	0	2
printf()	3	20	2
while()	1	"%d"	3
scanf()	3	"How many numbers you are going to enter:"	1
for(;;)	1	"\nEnter the numbers one by one:"	1
++	1	"\nThis is a test"	1
;	8	number	1
sort_numbers_ascending()	1	Calling the Function	1
/* */	1		
=	2		
&	3		
,	7		
>	1		
<	1		
{}	2		
n1=18	N1=40	n2=11	N2=23

3η ρουτίνα

```
/*
 *
 *
My code
 *
 *
*/
#include <stdio.h>
void main()
{
   int i, num[20], t=0;
   int n, count, j, a, x, b;

   printf("How many numbers you are going to enter:");
   scanf("%d", &count);
   printf("\nEnter the numbers one by one:");
/*
*
*
*
Test this code
*
*
*
*/
   while (t>20)
   {
/*test*/
    printf("\nThis is a test");
    scanf("%d", &count);
    printf("\nThis is my test");
    scanf("%d", &count);
   }

for(t=20; t<20; t--)
{
scanf("%d", &count);
}
/*My loop begins*/
   for (i = 0; i < count; ++i)
      scanf("%d", &num[i]);

 for (i = 0; i < n; ++i){
   for (j = i + 1; j < n; ++j){
      if (num[i] > num[j]){
         a = num[i];
         num[i] = num[j];
         num[j] = a;
      }
   }
}
/*Here are the data*/
  printf("Numbers in ascending order:\n");
   for (i = 0; i < count; ++i)
      printf("%d\n", num[i]);
}
```

Παραδοχές

1. Παίρνουμε την συνάρτηση main() μαζί με τις παρενθέσεις σαν έναν τελεστή στον ορισμό της, καθώς δεν γίνεται να οριστεί ορθά χωρίς αυτές.
2. Δεν συμπεριλαμβάνουμε όλα τα σχόλια σαν έντελα ή τελεστές, καθώς δεν αποτελούν μέρος του κώδικα. Μόνο όσα είναι εντός του κώδικα συμπεριλαμβάνονται.
3. Δεν συμπεριλαμβάνουμε το include σαν τελεστή ή σαν έντελο, καθώς βρίσκεται εκτός της ρουτίνας.
4. Παίρνουμε τον τελεστή for(; ;) σαν έναν, καθώς ο τελεστής for στην c εμφανίζεται υποχρεωτικά με ζεύγος παρενθέσεων και 2 ερωτηματικά.

Τελεστές	Αριθμός εμφανίσεων	Έντελα	Αριθμός εμφανίσεων
void	1	i	14
main()	1	20	4
int	2	0	4
while()	1	j	7
printf()	6	a	3
scanf()	5	b	1
for(;;)	5	x	1
If()	1	count	7
num[]	8	"How many numbers you are going to enter:"	1
=	8	"%d"	5
,	13	"\nEnter the numbers one by one:"	1
;	16	"\nThis is a test"	1
<	5	"\nThis is my test"	1
>	2	"Numbers in ascending order:\n"	1
++	3	"%d\n"	1
+	1	t	4
--	1	...**test this code..	1
/* */	4	test	1
{}	5	My loop…	1
&	5	Here are..	1
		n	3
n1=20	N1=93	n2=21	N2=63

ΕΡΩΤΗΜΑ 2^{Ο}

1. Λόγος του εκτιμητή μήκους προς το μήκος προγράμματος του Halstead (Nest/N)

   Έχουμε για το μήκος προγράμματος: N = N1 + N2

   Και για τον Εκτιμητή Μήκους: Nest = n1 $\log_2$n1 + n2 $\log_2$n2

   Άρα,

   <u>Για την 1^{η} ρουτίνα:</u>

   N=47+27=74, Nest= 15 $\log_2$15+ 8 $\log_2$8= 15*3.9+ 8*3= 58.5+24= 82.5

   Άρα, Nest/N = 1.11

   <u>Για την 2^{η} ρουτίνα:</u>

   N=40+23=63, Nest= 18 $\log_2$18+ 11 $\log_2$11= 113.56

   Άρα, Nest/N = 1.8

   <u>Για την 3^{η} ρουτίνα:</u>

   N=93+63=156, Nest= 20 $\log_2$20+ 21 $\log_2$21= 178.67

2. Επίπεδο προγράμματος του Halstead (L)

Κανονικά χρησιμοποιείται ο τύπος L = V*/ V. Όμως, στην προκειμένη ο υπολογισμός του δυναμικού όγκου V* δεν είναι εύκολος, καθώς πρέπει να βρούμε την πιο συνοπτική θεωρητική υλοποίηση του προγράμματος. Για αυτό χρησιμοποιούμε τον τύπο που μας δίνεται για την εκτίμηση του L, Lest, ο οποίος είναι ο εξής: Lest= 2n2/n1N2

Άρα,

<u>Για την 1^{η} ρουτίνα:</u>

Lest= 2*8/15*27= 0.0395

<u>Για την 2^{η} ρουτίνα:</u>

Lest=2*11/18*23= 0.0531

<u>Για την 3η ρουτίνα:</u>

Lest=2*21/20*63= 0.033

3. Επίπεδο γλώσσας του Halstead (λ)

Ισχύει ότι η γλώσσα υπολογίζεται ως λ = LV* = L^2V, όπου L είναι το επίπεδο προγράμματος που υπολογίσαμε παραπάνω και V είναι ο όγκος που υπολογίζεται ως V = N $\log_2$n.

<u>Για την 1^{η} ρουτίνα:</u>

N=74, n=15+8=23

Άρα, V=74 $\log_2$23= 74*4.5= 333

Άρα, τελικά έχουμε λ= $(0.0395)^2$*333=0.5(=0.52)

Για την 2η ρουτίνα:

N=63, n=18+11= 29

Άρα, V=63 $\log_2 29$= 63*4.9=308.7

Άρα, τελικά έχουμε λ=$(0.0531)^2$*308.7=0.87

Για την 3η ρουτίνα:

N=156, n=20+21= 41

Άρα, V=156 $\log_2 41$= 156*5.36= 836.16

Άρα, τελικά έχουμε λ=$(0.033)^2$*836.16=0.91

4. Λόγος αριθμού γραμμών σχολίων προς τον αριθμό φυσικών γραμμών κώδικα (Lines of Comments / Physical Lines of Code).

Για την 1η ρουτίνα:

Lines of comments=1, μόνο όσα σχόλια είναι εντός κώδικα

Physical Lines of Code=21, συμπεριλαμβανομένου κενές γραμμές, σχόλια μετά το include.

Χωρίς να συμπεριλαμβάνουμε το include και ό,τι βρίσκεται πάνω από αυτό.

(Lines of Comments / Physical Lines of Code)=1/21=0.048

Για την 2η ρουτίνα:

Lines of comments=1

Physical Lines of Code=18

(Lines of Comments / Physical Lines of Code)=1/18=0.055

Για την 3η ρουτίνα:

Lines of comments=12, μόνο όσα σχόλια είναι εντός κώδικα

Physical Lines of Code=50, συμπεριλαμβανομένου κενές γραμμές, σχόλια μετά το include.

Χωρίς να συμπεριλαμβάνουμε το include και ό,τι βρίσκεται πάνω από αυτό.

(Lines of Comments / Physical Lines of Code)=12/50=0.24

ΕΡΩΤΗΜΑ 3^{Ο}

Σ1. Ουσιαστικά, παίρνουμε για την κάθε ρουτίνα την τιμή της σε κάθε μετρική και διαιρούμε το άθροισμα των τιμών με το 2, καθώς έχουμε υπολογισμό μέσου όρου.

Nest/N (1^{η} μετρική)

1^{η} ρουτίνα: Nest1/N1 = 1.11

2^{η} ρουτίνα: Nest2/N2 = 1.8

Άρα, έχουμε για τον Μ.Ο. της 1^{ης} μετρικής: (1.11+1.8)/2=2.91/2= 1.455

L: Επίπεδο προγράμματος του Halstead (2^{η} μετρική)

1^{η} ρουτίνα: Lest= 0.0395

2^{η} ρουτίνα: Lest= 0.0531

Άρα, έχουμε για τον Μ.Ο. της 2^{ης} μετρικής: (0.0395+0.0531)/2= 0.0463

λ: Επίπεδο γλώσσας του Halstead (3^{η} μετρική)

1^{η} ρουτίνα: λ=0.52

2^{η} ρουτίνα: λ=0.87

Άρα, έχουμε για τον Μ.Ο. της 3^{ης} μετρικής: (0.52+0.87)/2= 0.695

Lines of Comments / Physical Lines of Code (4^{η} μετρική)

1^{η} ρουτίνα: (Lines of Comments / Physical Lines of Code)=1/21=0.048

2^{η} ρουτίνα: (Lines of Comments / Physical Lines of Code)=1/18=0.055

Άρα, έχουμε για τον Μ.Ο. της 4^{ης} μετρικής: (0.048+0.055)/2= 0.0515

Σ2. Ουσιαστικά, παίρνουμε για την κάθε ρουτίνα τις τιμές της και τις πολλαπλασιάζουμε με τα N της 1^{ης} και N της 2^{ης} ρουτίνας αντίστοιχα και τα προσθέτουμε και στο τέλος διαιρούμε το συνολικό άθροισμα με το άθροισμα των N1, N2.

1^{η} ρουτίνα: N1=47, N2=27, άρα Νολ1= 74

2^{η} ρουτίνα: N1=40, N2=23, άρα Νολ2= 63

Nest/N (1^{η} μετρική)

1^{η} ρουτίνα: Nest1/N1 = 1.11, άρα (Nest1/N1)*Νολ1= 82.14

2^{η} ρουτίνα: Nest2/N2 = 1.8, άρα (Nest2/N2)*Νολ2= 113.4

Άρα, έχουμε για τον σταθμισμένο Μ.Ο. της 1^{ης} μετρικής: (82.14+113.4)/(Νολ1+Νολ2)= 1.43

L: Επίπεδο προγράμματος του Halstead (2^{η} μετρική)

1^{η} ρουτίνα: Lest= 0.0395, άρα Lest*Νολ1= 2.92

2^{η} ρουτίνα: Lest= 0.0531, άρα Lest*Νολ2= 3.35

Άρα, έχουμε για τον σταθμισμένο Μ.Ο. της 2ης μετρικής: (2.92+3.35)/(Νολ1+Νολ2)= 0.0458

λ: Επίπεδο γλώσσας του Halstead (3η μετρική)

1η ρουτίνα: λ=0.52, άρα λ*Νολ1= 0.52*74= 38.48

2η ρουτίνα: λ=0.87, άρα λ*Νολ2=0.87*63= 54.81

Άρα, έχουμε για τον σταθμισμένο Μ.Ο. της 3ης μετρικής: (38.48+54.81)/(Νολ1+Νολ2)=0.68

Lines of Comments / Physical Lines of Code (4η μετρική)

1η ρουτίνα: (Lines of Comments / Physical Lines of Code)=1/21=0.048, άρα

(Lines of Comments / Physical Lines of Code)*Νολ1= 3.552

2η ρουτίνα: (Lines of Comments / Physical Lines of Code)=1/18=0.055, άρα

(Lines of Comments / Physical Lines of Code)*Νολ2= 3.465

Άρα, έχουμε για τον σταθμισμένο Μ.Ο. της 4ης μετρικής: (3.552+3.465)/(Νολ1+Νολ2)= 0.0512

Το καταλληλότερο σενάριο είναι το 2ο από τα παραπάνω, καθώς ο σταθμισμένος μέσος όρος των μετρικών είναι πιο ακριβής από τον απλό μέσο όρο τους, με βάση τον τρόπου υπολογισμού του. Αναλυτικότερα, με τον πολλαπλασιασμό κάθε ρουτίνας με το Νολ και με την αντίστοιχη τελική διαίρεση με το άθροισμα των Νολ όλων των ρουτινών επιτυγχάνεται ένα αποτέλεσμα πιο κοντινό στο πραγματικό. Με τον υπολογισμό του απλού μέσου όρου το σφάλμα είναι αρκετά μεγάλο, καθώς οι ρουτίνες ανεξάρτητα του πλήθους τελεστών τους έχουν την ίδια βαρύτητα στο τελικό αποτέλεσμα, πράγμα που διορθώνεται με τον σταθμισμένο μέσο όρο και την σωστή χρήση των Ν ως συντελεστές με τον τρόπο που αναφέρθηκε. Το παραπάνω μπορούμε να το συνειδητοποιήσουμε αν συγκρίνουμε τις τιμές του σταθμισμένου μέσου όρου με τις πραγματικές τιμές των μετρικών της υλοποίησης, οι οποίες φαίνονται παρακάτω. Πράγματι, είναι πιο κοντά στις πραγματικές τιμές των μετρικών οι σταθμισμένοι μέσοι όροι συγκριτικά με τους απλούς μέσους όρους.

Υλοποίηση Α΄

Τελεστές	Αριθμός εμφανίσεων	Έντελα	Αριθμός εμφανίσεων
void	2	count	7
int	4	i	8
number[]	9	j	7
,	12	k	7
sort_numbers_ascending()	2	temp	3
for(; ;)	4	"Numbers in ascending order:\n"	1
if()	1	"%d\n"	1
{}	6	0	4
printf()	5	20	2
<	4	t	2
>	2	"%d"	3
=	8	"How many numbers you are going to enter:"	1
;	14		
++	4		
+	1		
scanf()	3		
/* */	1	"\nEnter the numbers one by one:"	1
while()	1	"\nThis is a test"	1
&	3	number	1
		Calling the Function	1
n1=19	N1=85	n2=16	N2=50

Παραδοχές

1. Παίρνουμε την συνάρτηση sort_numbers_ascending() μαζί με τις παρενθέσεις σαν έναν τελεστή στον ορισμό της, καθώς δεν γίνεται να οριστεί ορθά χωρίς αυτές.
2. Δεν συμπεριλαμβάνουμε όλα τα σχόλια σαν έντελα ή τελεστές, καθώς δεν αποτελούν όλα μέρος του κώδικα. Προσμετρούνται μόνο όσα είναι εντός του κώδικα.
3. Δεν συμπεριλαμβάνουμε το include σαν τελεστή ή σαν έντελο, και ο,τι βρίσκεται πάνω από αυτό.
4. Παίρνουμε τον τελεστή for(; ;) σαν έναν, καθώς ο τελεστής for στην c εμφανίζεται υποχρεωτικά με ζεύγος παρενθέσεων και 2 ερωτηματικά.

Μετρικές

1. Έχουμε για το μήκος προγράμματος: N = N1 + N2= 85+50= 135

Και για τον Εκτιμητή Μήκους: Nest = n1 $\log_2$n1 + n2 $\log_2$n2= 19*4.248 + 16*4= 144.712

Άρα, έχουμε Nest/ N = 144.712/135= 1.07

2. Έχουμε για το επίπεδο προγράμματος Lest= 2n2/n1N2= 2*16/19*50= 0.033

3. Έχουμε για το επίπεδο γλώσσας του Halstead (λ) ότι $\lambda = LV^* = L^2V$, όπου L είναι το επίπεδο προγράμματος που υπολογίσαμε παραπάνω και V είναι ο όγκος που υπολογίζεται ως V = N $\log_2$n, άρα $\lambda=(0.033)^2*135*5.129= 0.75$

4. Έχουμε για την τιμή του (Lines of Comments / Physical Lines of Code)=2/39= 0.0512

ΕΡΩΤΗΜΑ 4^{Ο}

Όπως προαναφέρθηκε, πράγματι το 2^{ο} σενάριο είναι το πιο ακριβές για υπολογισμό των μετρικών, λόγω της στάθμισης της τιμής του μέσου όρου. Έτσι, για την υλοποίηση Α΄ με βάση το 2^{ο} σενάριο έχουμε τις εξής τιμές:

Nest/N (1^{η} μετρική)

Σταθμισμένος Μ.Ο. της 1^{ης} μετρικής για Α' υπόθεση: (82.14+113.4)/(Νολ1+Νολ2)= 1.43

Τιμή 1^{ης} μετρικής για Β΄ υπόθεση: (Nest/N)= 1.14

Στον υπολογισμό του Nest έχουμε για n στοιχεία την πράξη n στοιχεία * $\log_2$ n, το οποίο ισούται με τις ιδανικές φορές που θα χρειαστεί να αναφερθεί στον ιδεατό κώδικα κάθε έντελο και κάθε τελεστής. Άρα, στις υλοποιήσεις που πλησιάζουν το ιδανικό η τιμή του Nest ισούται με την τιμή του Ν, συνεπώς ο λόγος Nest/N στην ιδανική περίπτωση θα είναι 1. Συμπερασματικά, συγκρίνοντας τις παραπάνω τιμές, πιο κοντά στην μονάδα, άρα και καλύτερη μετρική αποτελεί εκείνη της Β΄ υλοποίησης.

Παρολαυτά, σημαντικό είναι να σημειωθεί ότι το κλάσμα Nest/N βγάζει σωστά αποτελέσματα για εύλογου μεγέθους κώδικα(δηλαδή περίπου 500 γραμμές) και όχι σε μικρά προγράμματα όπως των υλοποιήσεων που μας δίνονται.

L: Επίπεδο προγράμματος του Halstead (2^{η} μετρική)

Σταθμισμένος Μ.Ο. της 2^{ης} μετρικής για Α' υπόθεση: (2.92+3.35)/(Νολ1+Νολ2)= 0.0458

Τιμή 2^{ης} μετρικής για Β΄ υπόθεση: Lest=2*21/20*63= 0.033

Ομοίως με την 1^{η} μετρική, έχουμε για το επίπεδο προγράμματος L, που βρίσκουμε προσεγγιστικά με το Lest, ότι ισχύει L=V*/V. Άρα για την καλύτερη θεωρητική υλοποίηση, η οποία θα έχει το υψηλότερο επίπεδο θα έχουμε L=1. Όσο μικραίνει η τιμή του L και αποκλίνει από την μονάδα, τόσο χαμηλότερου επιπέδου υλοποίηση έχουμε.

Συμπερασματικά, για τις παραπάνω υλοποιήσεις πιο κοντά στην μονάδα είναι η τιμή της Α' υλοποίησης, άρα εκείνη έχει και το υψηλότερο επίπεδο.

λ: Επίπεδο γλώσσας του Halstead (3^{η} μετρική)

Σταθμισμένος Μ.Ο. της 3^{ης} μετρικής για Α΄ υπόθεση: (38.48+54.81)/(Νολ1+Νολ2)=0.68

Τιμή 3^{ης} μετρικής για Β΄ υπόθεση: λ=$(0.033)^2$*836.16=0.91

Όσο πιο μεγάλο είναι το λ, τόσο πιο abstract είναι η γλώσσα, δηλαδή τόσο πιο υψηλού επιπέδου είναι. Αν συγκρίνουμε το γνωστό λ της c με τα λ που υπολογίσαμε μπορούμε να συμπεράνουμε ότι η υλοποίηση που έχει λ πιο κοντά σε αυτό είναι πιο κοντά στην ιδανική.

Δεν έχουμε την μετρική λ της γλώσσας που είναι γραμμένα τα προγράμματα, όμως αν συγκρίνουμε τις τιμές των 2 υλοποιήσεων μεγαλύτερο λ έχει η Β΄ υλοποίηση, άρα εκείνη είναι υψηλότερου επιπέδου.

(Δηλαδή, η μετρική επιπέδου γλώσσας εξαρτάται κατά κύριο λόγο από την γλώσσα προγραμματισμού, που στην προκειμένη έχουμε και στις 2 υλοποιήσεις την ίδια γλώσσα. Η διαφορά τους οφείλεται στην πολυπλοκότητα του κάθε προγράμματος.)

Lines of Comments / Physical Lines of Code (4η μετρική)

Σταθμισμένος Μ.Ο. της 4ης μετρικής για Α΄ υπόθεση: (3.552+3.465)/(Νολ1+Νολ2)= 0.0512

Τιμή 4ης μετρικής για Β΄ υπόθεση: (Lines of Comments / Physical Lines of Code)=12/50=0.24

Το κλάσμα γραμμών σχολίων προς φυσικές γραμμές κώδικα δεν αποτελεί μέτρο για την ποιότητα του κώδικα, καθώς δεν λαμβάνει υπόψη την πολυπλοκότητα ή την λειτουργία του, αλλά ούτε και το μέγεθος των προδιαγραφών του, παρά μόνο συγκρίνει τον αριθμό των σειρών. Ωστόσο, το PLOC είναι σημαντικό για εκτίμηση του πλήθους των γραμμων κώδικα.

Άρα, συνολικά από τις παραπάνω μετρικές προκύπτει ότι η Β΄ υλοποίηση είναι καλύτερη από την Α΄ υλοποίηση καθώς οι τιμές των μετρικών της είναι πιο κοντά σε εκείνες των ιδανικών μετρικών.