ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
UNIVERSITY OF WEST ATTICA

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΡΓΑΣΙΑ 5
ΥΠΟΠΡΟΓΡΑΜΜΑΤΑ

ΣΤΟΙΧΕΙΑ ΦΟΙΤΗΤΗ:

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: ΑΘΑΝΑΣΙΟΥ ΒΑΣΙΛΕΙΟΣ ΕΥΑΓΓΕΛΟΣ
ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ: 19390005
ΚΑΤΑΣΤΑΣΗ ΦΟΙΤΗΤΗ: ΠΡΟΠΤΥΧΙΑΚΟ
ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ: ΠΑΔΑ
ΤΜΗΜΑ ΕΡΓΑΣΤΗΡΙΟΥ: Μ2
ΚΑΘΗΓΗΤΗΣ ΕΡΓΑΣΤΗΡΙΟΥ: ΓΕΩΡΓΙΟΣ ΜΕΛΕΤΙΟΥ
ΗΜΕΡΟΜΗΝΙΑ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ: 21/12/2021

ΠΗΓΑΙΟΙ ΚΩΔΙΚΕΣ / ΤΕΚΜΗΡΙΩΣΕΙΣ

ΘΕΜΑ 1

ΕΠΙΣΗΜΑΝΣΗ «SinCosTaylor.c»

Το «Πρόγραμμα “SinCosTaylor.c”» (Πηγαίος Κώδικας) και η «Τεκμηρίωση “ SinCosTaylor.c”» (Ζητούμενο, Δομή, Συναρτήσεις, Μεταβλητές, Διάσχιση, Παραδείγματα, Παρατηρήσεις) απαντούν στο ζητούμενο του ερωτήματος «Θέμα 1».

ΠΡΟΓΡΑΜΜΑ « SinCosTaylor.c»

```
1  #include <stdio.h>
2  #include <math.h>
3  #define pi 3.14159
4  /* Δήλωση συναρτήσεων */
5  void Title (); // Ο τίτλος του προγράμματος
6  double Read_Deg (); // Εισαγωγή της γωνίας σε μοίρες
7  double Deg_to_Rad (double); // Μετατροπή της γωνίας από μοίρες σε ακτίνια
8  void Print_Deg (double); // Εκτύπωση της γωνίας σε μοίρες
9  void Print_Rad (double); // Εκτύπωση της γωνίας σε ακτίνια
10 double Sin (double); // Υπολογισμός του ημιτόνου της γωνίας με τη συνάρτηση "sin (ω)"
11 double Taylor_S (double); // Υπολογισμός του ημιτόνου της γωνίας με την απειροσειρά "Taylor"
12 void Print_Sin_TaylorS (double); // Εκτύπωση του ημιτόνου της γωνίας με τη συνάρτηση "sin (ω)" και με την απειροσειρά "Taylor"
13 int Check_Sin_TaylorS (double); // Σύγκριση των συναρτήσεων "Sin (ω)" και "Taylor_S (ω)" που υπολογίζουν το ημίτονο της γωνίας σε ακτίνια για το αν είναι "σχεδόν" ίσοι
14 double Cos (double); // Υπολογισμός του συνημιτόνου της γωνίας με τη συνάρτηση "cos (ω)"
15 double Taylor_C (double); // Υπολογισμός του συνημιτόνου της γωνίας με την απειροσειρά "Taylor"
16 void Print_Cos_TaylorC (double); // Εκτύπωση του συνημιτόνου της γωνίας με τη συνάρτηση "cos (ω)" και με την απειροσειρά "Taylor"
17 int Check_Cos_TaylorC (double); // Σύγκριση των συναρτήσεων "Cos (ω)" και "Taylor_C (ω)" που υπολογίζουν το συνημίτονο της γωνίας σε ακτίνια για το αν είναι "σχεδόν" ίσοι
18 /* Όπου "ω" η γωνία σε ακτίνια και "Ω" η γωνία σε μοίρες */
19
20  int main (int argc, char **argv) /* main (int argc, char **argv) */
```

```
21  {
22     system ("chcp 1253");
23
24     double deg, rad; // Δήλωση μεταβλητών
25
26     Title (); // Κλήση της συνάρτησης "Title ()"
27     deg = Read_Deg (); // Κλήση της συνάρτησης "Read_Deg ()"
28     rad = Deg_to_Rad (deg); // Κλήση της συνάρτησης "Deg_to_Rad (Ω)"
29     Print_Deg (deg); // Κλήση της συνάρτησης "Print_Deg (Ω)"
30     Print_Rad (rad); // Κλήση της συνάρτησης "Print_Rad (ω)"
31     Print_Sin_TaylorS (rad); // Κλήση της συνάρτησης "Print_Sin_TaylorS
(ω)"
32     Check_Sin_TaylorS (rad); // Κλήση της συνάρτησης "Check_Sin_TaylorS
(ω)"
33     Print_Cos_TaylorC (rad); // Κλήση της συνάρτησης "Print_Cos_TaylorC
(ω)"
34     Check_Cos_TaylorC (rad); // Κλήση της συνάρτησης "Check_Cos_TaylorC
(ω)"
35
36     return 0;
37  }
38
39  void Title () /* Title () */
40  {
41     printf ("============================================================
\n\n");
42     printf ("Υπολογισμός του ημιτόνου και του συνημιτόνου μίας γωνίας\n\n");
// Τίτλος προγράμματος
43     printf ("============================================================
\n\n");
44  }
45
46  double Read_Deg () /* Read_Deg () */
47  {
48     double deg_RD; // Δήλωση μεταβλητών
49
50     printf ("Εισάγετε γωνία στο διάστημα του 1ου κύκλου [0,360]\n\n");
51     printf ("Μοίρες  : ");
52     scanf ("%lf", &deg_RD); // Εισαγωγή της γωνίας σε μοίρες
```

```
53     printf     ("\n-------------------------------------------------------
\n\n");
54
55     return deg_RD; // Επιστροφή της γωνίας σε μοίρες
56   }
57
58   double Deg_to_Rad (double deg_DtR) /* Deg_to_Rad (Ω) */
59   {
60     double rad_dtr; // Δήλωση μεταβλητών
61
62     rad_dtr = (pi * deg_DtR) / 180; // Μετατροπή της γωνίας από μοίρες σε
ακτίνια
63
64     return rad_dtr; // Επιστροφή της γωνίας σε ακτίνια
65   }
66
67   void Print_Deg (double deg_PD) /* Print_Deg (Ω) */
68   {
69     printf ("Μοίρες   : [%20.6lf]\n", deg_PD); // Εκτύπωση της γωνίας σε
μοίρες
70   }
71
72   void Print_Rad (double rad_PR) /* Print_Rad (ω) */
73   {
74     printf ("Ακτίνια : [%20.6lf]\n\n", rad_PD); // Εκτύπωση της γωνίας σε
ακτίνια
75   }
76
77   double Sin (double rad_S) /* Sin (ω) */
78   {
79     double c; // Δήλωση μεταβλητών
80
81     c = sin (rad_S); // Υπολογισμός του ημιτόνου της γωνίας με τη συνάρτηση
"sin (ω)"
82
83     return c; // Επιστροφή του ημιτόνου της γωνίας με τη συνάρτηση "sin
(ω)"
84 }
85
```

```
86  double Taylor_S (double rad_TaylorS) /* Taylor_S (ω) */
87  {
88     double  term,  next_term,  diff_terms,  abs_diff_terms,  first_sin_T;
// Δήλωση μεταβλητών
89     double sin_T = 0.0; // Αρχικοποίηση μεταβλητών
90     int i;
91     int sign = -1; // Αρχικοποίηση μεταβλητών
92     int j = 1; // Αρχικοποίηση μεταβλητών
93
94     do /* 1ος Βρόχος */
95     {
96           term = 1; // Αρχικοποίηση μεταβλητής του πρώτου όρου, του δευτέρου
...
97           for (i = 1 ; i <= j ; i++) /* 2ος Βρόχος */
98           {
99                 term = term * (rad_TaylorS / i); // Υπολογισμός του πρώτου
όρου, του δευτέρου ... (ω^1 / 1!, ω^3 / 3! ...)
100          }
101          j = j + 2; // Αύξηση της βοηθητικής μεταβλητής για τον υπολογισμό
του δευτέρου όρου, του τρίτου ...
102          next_term =  term * ((rad_TaylorS * rad_TaylorS) / (j * (j -
1))); // Υπολογισμός του δευτέρου όρου, του τρίτου ... (ω^3 / 3!, ω^5 / 5!
...)
103          diff_terms = next_term - term; // Υπολογισμός της διαφοράς του
δευτέρου όρου με τον πρώτο, του τρίτου με τον δεύτερο ...
104          abs_diff_terms = fabs (diff_terms); // Υπολογισμός της απόλυτης
τιμής της διαφοράς των όρων
105          if (j == 3) /* (~) 1η επανάληψη του 1ου βρόχου */
106          {  /* ω^1 / 1! - ω^3 / 3! */
107                first_sin_T = term + (sign * next_term); // Υπολογισμός
του πρώτου αθροίσματος με το κατάλληλο πρόσημο "sign"
108                sin_T = sin_T + first_sin_T; // Καταχώρηση του αθροίσματος
στη μεταβλητή "sin_T"
109                sign = sign * (-1); // Αλλαγή προσήμου
110          }
111          else /* (~) 2η, 3η ... επανάληψη του 1ου βρόχου */
112          {  /* (ω^1 / 1! - ω^3 / 3!) + ω^5 / 5! ... */
113                sin_T = sin_T + (sign * next_term); // Υπολογισμός του
δεύτερου αθροίσματος, του τρίτου ... με το κατάλληλο πρόσημο "sign"
114                sign = sign * (-1); // Αλλαγή προσήμου
115          }
```

```
116    }
117    while (abs_diff_terms > 0.000001);
118
119    return sin_T; // Επιστροφή του αθροίσματος των όρων (απειροσειρά Taylor)
120 }
121
122 void Print_Sin_TaylorS (double rad_PSTS) /* Print_Sin_TaylorS (ω) */
123 {
124    double rad_Sin, rad_TaylorSin; // Δήλωση μεταβλητών
125
126    rad_Sin = Sin (rad_PSTS); // Κλήση της συνάρτησης "Sin (ω)"
127    printf ("Sine    : [%20.6lf]\n", rad_Sin); // Εκτύπωση του ημιτόνου
της γωνίας με την έτοιμη συνάρτηση "sin (ω)"
128    rad_TaylorSin = Taylor_S (rad_PSTS); // Κλήση της συνάρτησης "Taylor_S
(ω)"
129    printf ("Taylor  : [%20.6lf]\n\n", rad_TaylorSin); // Εκτύπωση του
ημιτόνου της γωνίας με την απειροσειρά "Taylor"
130 }
131
132 int Check_Sin_TaylorS (double rad_CSTS) /* Check_Sin_TaylorS (ω) */
133 {
134    double rad_CheckSin, rad_CheckTaylorS;
135    double diff_CheckSinTaylorS, abs_diff_CheckSinTaylorS; // Δήλωση
μεταβλητών
136
137    rad_CheckSin = Sin (rad_CSTS); // Κλήση της συνάρτησης "Sin (ω)"
138    rad_CheckTaylorS = Taylor_S (rad_CSTS); // Κλήση της συνάρτησης
"Taylor_S (ω)"
139    diff_CheckSinTaylorS = rad_CheckSin - rad_CheckTaylorS; // Υπολογισμός
της διαφοράς της συνάρτησης "sin (ω)" με την απειροσειρά "Taylor"
140    abs_diff_CheckSinTaylorS = fabs (diff_CheckSinTaylorS); // Υπολογισμός
της απόλυτης τιμής της διαφοράς της συνάρτησης "sin (ω)" με την απειροσειρά
"Taylor"
141    if (abs_diff_CheckSinTaylorS <= 0.0000009) /* (~) Αποδεκτή απόλυτη
τιμή της διαφοράς */
142    {
143          printf ("Sine ~= Taylor\n");
144          printf ("Οι δύο αριθμοί είναι σχεδόν ίσοι\n\n");
145    }
146    else  /* (~) Απορριπτέα απόλυτη τιμή της διαφοράς */
```

```
147    {
148            printf ("Sine != Taylor\n");
149            printf ("Οι δύο αριθμοί δεν είναι σχεδόν ίσοι\n\n");
150    }
151 }
152
153 double Cos (double rad_C) /* Cos (ω) */
154 {
155    double d; // Δήλωση μεταβλητών
156
157    d = cos (rad_C); // Υπολογισμός του συνημιτόνου της γωνίας με τη συνάρτηση "cos (ω)"
158
159    return d; // Επιστροφή του συνημιτόνου της γωνίας με τη συνάρτηση "cos (ω)"
160 }
161
162 double Taylor_C (double rad_TaylorC) /* Taylor_C (ω) */
163 {
164    double term, next_term, diff_terms, abs_diff_terms, first_cos_T; // Δήλωση μεταβλητών
165    double cos_T = 0.0; // Αρχικοποίηση μεταβλητών
166    int i;
167    int sign = -1;
168    int j = 0;
169
170    do /* 1ος Βρόχος */
171    {
172            term = 1; // Αρχικοποίηση μεταβλητής του πρώτου όρου, του δευτέρου ...
173            for (i = 1 ; i <= j ; i++) /* 2ος Βρόχος */
174            {
175                    term = term * (rad_TaylorC / i); // Υπολογισμός του πρώτου όρου, του δευτέρου ... (1, ω^2 / 2! ...)
176            }
177            j = j + 2; // Αύξηση της βοηθητικής μεταβλητής για τον υπολογισμό του δευτέρου όρου, του τρίτου ...
178            next_term =  term * ((rad_TaylorC * rad_TaylorC) / (j * (j - 1))); // Υπολογισμός του δευτέρου όρου, του τρίτου ... (ω^2 / 2!, ω^4 / 4! ...)
```

```
179          diff_terms = next_term - term; // Υπολογισμός της διαφοράς του δευτέρου όρου με τον πρώτο, του τρίτου με τον δεύτερο ...
180          abs_diff_terms = fabs (diff_terms); // Υπολογισμός της απόλυτης τιμής της διαφοράς των όρων
181          if (j == 2) /* (~) 1η επανάληψη του 1ου βρόχου */
182          {  /* 1 - ω^2 / 2! */
183                  first_cos_T = term + (sign * next_term); // Υπολογισμός του πρώτου αθροίσματος με το κατάλληλο πρόσημο "sign"
184                  cos_T = cos_T + first_cos_T; // Καταχώρηση του αθροίσματος στη μεταβλητή "cos_T"
185                  sign = sign * (-1); // Αλλαγή προσήμου
186          }  /* (1 - ω^2 / 2!) + ω^4 / 4! ... */
187          else /* (~) 2η, 3η ... επανάληψη του 1ου βρόχου */
188          {
189                  cos_T = cos_T + (sign * next_term); // Υπολογισμός του αθροίσματος του δευτέρου αθροίσματος, του τρίτου ... με το κατάλληλο πρόσημο "sign"
190                  sign = sign * (-1); // Αλλαγή προσήμου
191          }
192   }
193   while (abs_diff_terms > 0.000001);
194
195   return cos_T; // Επιστροφή του αθροίσματος των όρων (απειροσειρά Taylor)
196 }
197
198 void Print_Cos_TaylorC (double rad_PCTC) /* Print_Cos_TaylorC (ω) */
199 {
200   double rad_Cos, rad_TaylorCos; // Δήλωση μεταβλητών
201
202   rad_Cos = Cos (rad_PCTC); // Κλήση της συνάρτησης "Cos (ω)"
203   printf ("Cosine  : [%20.6lf]\n", rad_Cos); // Εκτύπωση του συνημιτόνου της γωνίας με τη συνάρτηση "cos (ω)"
204   rad_TaylorCos = Taylor_C (rad_PCTC); // Κλήση της συνάρτησης "Taylor_C (ω)"
205   printf ("Taylor  : [%20.6lf]\n\n", rad_TaylorCos); // Εκτύπωση του συνημιτόνου της γωνίας με την απειροσειρά "Taylor"
206 }
207
208 int Check_Cos_TaylorC (double rad_CCTC) /* Check_Cos_TaylorC (ω) */
209 {
```

```
210    double rad_CheckCos, rad_CheckTaylorC;

211    double diff_CheckCosTaylorC, abs_diff_CheckCosTaylorC; // Δήλωση μεταβλητών

212

213    rad_CheckCos = Cos (rad_CCTC); // Κλήση της συνάρτησης "Cos (ω)"

214    rad_CheckTaylorC = Taylor_C (rad_CCTC); // Κλήση της συνάρτησης "Taylor_C (ω)"

215    diff_CheckCosTaylorC = rad_CheckCos - rad_CheckTaylorC; // Υπολογισμός της διαφοράς της συνάρτησης "cos (ω)" με την απειροσειρά "Taylor"

216   abs_diff_CheckCosTaylorC = fabs (diff_CheckCosTaylorC); // Υπολογισμός της απόλυτης τιμής της διαφοράς της συνάρτησης "cos (ω)" με την απειροσειρά "Taylor"

217    if (abs_diff_CheckCosTaylorC <= 0.0000009) /* (~) Αποδεκτή απόλυτη τιμή της διαφοράς */

218    {

219           printf ("Cosine ~= Taylor\n");

220           printf ("Οι δύο αριθμοί είναι σχεδόν ίσοι\n\n");

221    }

222   else /* (~) Απορριπτέα απόλυτη τιμή της διαφοράς */

223    {

224           printf ("Cosine != Taylor\n");

225           printf ("Οι δύο αριθμοί δεν είναι σχεδόν ίσοι\n\n");

226    }

227 }
```

ΤΕΚΜΗΡΙΩΣΗ « SinCosTaylor.c»

ΖΗΤΟΥΜΕΝΟ

Το πρόγραμμα «SinCosTaylor.c» επιτυγχάνει τις εξής λειτουργίες:

a) Διαβάζει από τη «standard» είσοδο μία γωνία «Ω» σε μοίρες.

b) Τη μετατρέπει σε ακτίνια (ω).

c) Υπολογίζει το ημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση «sin (ω)» της βιβλιοθήκης «math.h».

d) Υπολογίζει το ημίτονο της γωνίας σε ακτίνια με την χαρακτηριστική απειροσειρά «Taylor».

$$\sin(x)=x-\frac{x^3}{3!}+\frac{x^5}{5!}-\frac{x^7}{7!}+\frac{x^9}{9!}\cdots$$

e) Συγκρίνει τους δύο αριθμούς και καταλήγει στο πόρισμα αν είναι «σχεδόν» ίσοι ή όχι.

f) Υπολογίζει το συνημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση «cos (ω)» της βιβλιοθήκης «math.h».

g) Υπολογίζει το συνημίτονο της γωνίας σε ακτίνια με την χαρακτηριστική απειροσειρά «Taylor».

$$\cos(x)=1-\frac{x^2}{2!}+\frac{x^4}{4!}-\frac{x^6}{6!}+\frac{x^8}{8!}\cdots$$

h) Συγκρίνει τους δύο αριθμούς και καταλήγει στο πόρισμα αν είναι «σχεδόν» ίσοι ή όχι.

i) Τυπώνει τα αποτελέσματα απο τη «standard» έξοδο συνοδευόμενα με τα κατάλληλα μηνύματα.

ΔΟΜΗ

Προκειμένου να υλοποιηθεί το ζητούμενο χρησιμοποίηθηκαν, αρχικά, οι βιβλιοθήκες (.h) και οι εντολές :

a) «stdio.h»: Περιέχει τις έτοιμες συναρτήσεις «scanf(...)» και «printf(...)» που συνδέονται με τα κανάλια εισόδου και εξόδου αντίστοιχα για την ανάγνωση και την τύπωση περιεχομένων των αντίστοιχων μεταβλητών. Επίσης, η «printf(...)» χρησιμοποιήθηκε για να τυπωθούν χαρακτηριστικά μηνύματα για την βέλτιστη κατανόηση του πηγαίου κώδικα.

b) «math.h»: Περιέχει τις έτοιμες συνάρτησεις «sin(...)», «cos(...)» και «fabs(...)» για τον υπολογισμό του ημιτόνου και το συνημιτόνου μιας γωνίας αντίστοιχα, σε ακτίνια και της απόλυτης τιμής ενός αριθμού.

c) «define»: Στη γραμμή «3» ορίζεται ο αναγνωριστικός αριθμός «pi» και η σταθέρα «3.14159», όπου κατά την διάρκεια της προμεταγλώττισης του κώδικα θα αντικατασταθεί, όπου το «pi» η σταθερά «3.14159».

Επιπρόσθετα, χρησιμοποιήθηκαν οι χαρακτηριστικοί τελεστές :

a) αριθμητικοί : +, -, *, /

b) σχεσιακοί : <=, >, ==,

c) ανάθεσης : =

d) τελεστής & : Για την διεύθυνση μεταβλητής ως δεύτερο όρισμα της συνάρτησης «scanf()» που συνδέεται με τη «standard» είσοδο

e) μετααύξησης : μεταβλητή++
Οι εντολές ελέγχου :

a) if – else

Οι εντολές επαναλήψης :

a) do – while

b) for

Η κάθε λειτουργία απ' την ενότητα «Ζητούμενο» υλοποιήθηκε με αυτόνομα υποπρογράμματα (βλ. ενότητα «Συναρτήσεις»).

ΣΥΝΑΡΤΗΣΕΙΣ

Τύπου «void» (δεν επιστρέφουν τιμή)

Title () «Ο τίτλος του προγράμματος»

Print_Rad (rad_PR) «Εκτύπωση της γωνίας σε ακτίνια»

Print_Deg (deg_PD) «Εκτύπωση της γωνίας σε μοίρες»

Print_Sin_TaylorS (rad_PSTS) «Εκτύπωση του ημιτόνου της γωνίας με τη συνάρτηση "sin (ω)" και με την απειροσειρά "Taylor"»

Print_Cos_TaylorC (rad_PCTC) «Εκτύπωση του συνημιτόνου της γωνίας με τη συνάρτηση "cos (ω)" και με την απειροσειρά "Taylor"»

Check_Sin_TaylorS (rad_CSTS) «Σύγκριση των συναρτήσεων "Sin (ω)" και "Taylor_S (ω)" που υπολογίζουν το ημίτονο της γωνίας σε ακτίνια για το αν είναι "σχεδόν" ίσοι»

Check_Cos_TaylorC (rad_CCTC) «Σύγκριση των συναρτήσεων "Cos (ω)" και "Taylor_C (ω)" που υπολογίζουν το συνημίτονο της γωνίας σε ακτίνια για το αν είναι "σχεδόν" ίσοι»

Τύπου «int» (επιστρέφουν ακέραια τιμή)

main (int argc, char **argv) «Η κύρια συνάρτηση του προγράμματος»

Τύπου «double» (επιστρέφουν πραγματική τιμή διπλής ακρίβειας)

Read_Deg () «Εισαγωγή μίας γωνίας σε μοίρες»

Deg_to_Rad (deg_DtR) «Μετατροπή της γωνίας από μοίρες σε ακτίνια»

Sin (rad_S) «Υπολογισμός του ημιτόνου της γωνίας με τη συνάρτηση "sin (ω)"»

Taylor_S (rad_TS) «Υπολογισμός του ημιτόνου της γωνίας με την απειροσειρά "Taylor"»

Cos (rad_C) «Υπολογισμός του συνημιτόνου της γωνίας με τη συνάρτηση "cos (ω)"»

Taylor_C (rad_TC) «Υπολογισμός του συνημιτόνου της γωνίας με την απειροσειρά "Taylor"»

ΜΕΤΑΒΛΗΤΕΣ

main (int argc, char **argv)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

deg (Η γωνία σε μοίρες)

rad (Η γωνία σε ακτίνια)

Read_Deg ()

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

deg_RD (Η γωνία σε μοίρες)

Deg_to_Rad (deg_DtR)

Παράμετρος

deg_DtR (τύπου «double» / Η γωνία σε μοίρες)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

rad_dtr (Η γωνία σε ακτίνια)

Print_Deg (deg_PD)

Παράμετρος

deg_PD (τύπου «double» / Η γωνία σε μοίρες)

Print_Rad (rad_PR)

Παράμετρος

rad_PR (τύπου «double» / Η γωνία σε ακτίνια)

Sin (rad_S)

Παράμετρος

rad_S (τύπου «double» / Η γωνία σε ακτίνια)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

c (Το ημίτονο της γωνίας σε ακτίνια)

Taylor_S (rad_TS)

Παράμετρος

rad_TS (τύπου «double» / Η γωνία σε ακτίνια)

Ακέραιες μεταβλητές (τύπου «int»)

i (Ο μετρητής του δεύτερου βρόχου)

sign (Η βοηθητική μεταβλητή για την αλλαγή του προσήμου)

j (Η βοηθητική μεταβλητή για τον υπολογισμό του δευτέρου όρου, του τρίτου, ...)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

term (Ο πρώτος όρος, ο δεύτερος ...)

next_term (Ο δεύτερος όρος, ο τρίτος ...)

diff_terms (Η διαφορά του δεύτερου όρου με τον πρώτο, του τρίτου με τον

δεύτερο ...)

abs_diff_terms (Η απόλυτη τιμή της διαφοράς των όρων)

first_sin_T (Το άθροισμα του πρώτου με τον δεύτερο όρο με το κατάλληλο πρόσημο)

sin_T (Το άθροισμα του πρώτου με τον δεύτερο όρο, το άθροισμα αυτό με τον τρίτο όρο ... με το κατάλληλο πρόσημο)

Print_Sin_TaylorS (rad_PSTS)

Παράμετρος

rad_PSTS (τύπου «double» / Η γωνία σε ακτίνια)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

rad_Sin (Το ημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση)

rad_TaylorSin (Το ημίτονο της γωνίας σε ακτίνια με την απειροσειρά «Taylor»)

Check_Sin_TaylorS (rad_CSTS)

Παράμετρος

rad_CSTS (τύπου «double» / Η γωνία σε ακτίνια)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

CheckSin (Το ημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση)

CheckTaylorS (Το ημίτονο της γωνίας σε ακτίνια με την απειροσειρά «Taylor»)

diff_CheckSinTaylorS (Η διαφορά των δύο αριθμών που υπολογίζουν το ημίτονο της γωνίας σε ακτίνια)

abs_diff_CheckSinTaylorS (Η απόλυτη τιμή της διαφοράς)

Cos (rad_C)

Παράμετρος

rad_C (τύπου «double» / Η γωνία σε ακτίνια)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

d (Το ημίτονο της γωνίας σε ακτίνια)

Taylor_C (rad_TC)

Παράμετρος

rad_TC (τύπου «double» / Η γωνία σε ακτίνια)

Ακέραιες μεταβλητές (τύπου «int»)

i (βοηθητική μεταβλητή ελέγχου του δεύτερου βρόχου)

sign (βοηθητική μεταβλητή για την αλλαγή του προσήμου)

j (βοηθητική μεταβλητή για τον υπολογισμό του δευτέρου όρου, του τρίτου, ...)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

term (Ο πρώτος όρος, ο δεύτερος ...)

next_term (Ο δεύτερος όρος, ο τρίτος ...)

diff_terms (Η διαφορά του δεύτερου όρου με τον πρώτο, του τρίτου με τον δεύτερο ...)

abs_diff_terms (Η απόλυτη τιμή της διαφοράς)

first_cos_T (Το άθροισμα του πρώτου με τον δεύτερο όρο με το κατάλληλο πρόσημο)

cos_T (Το άθροισμα του πρώτου με τον δεύτερο όρο, το άθροισμα αυτό με τον με τον τρίτο όρο ... με το κατάλληλο πρόσημο)

Print_Cos_TaylorC (rad_PCTC)

Παράμετρος

rad_PCTC (τύπου «double» / Η γωνία σε ακτίνια)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

rad_Cos (Το συνημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση)

rad_TaylorCos (Το συνημίτονο της γωνίας σε ακτίνια με την απειροσειρά «Taylor»)

Check_Cos_TaylorC (rad_CCTC)

Παράμετρος

rad_CCTS (τύπου «double» / Η γωνία σε ακτίνια)

Πραγματικές μεταβλητές διπλής ακρίβειας (τύπου «double»)

CheckCos (Το συνημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση)

CheckTaylorC (Το συνημίτονο της γωνίας σε ακτίνια με την απειροσειρά «Taylor»)

diff_CheckCosTaylorC (Η διαφορά των δύο αριθμών που υπολογίζουν το συνημίτονο της γωνίας σε ακτίνια)

abs_diff_CheckCosTaylorC (Η απόλυτη τιμή της διαφοράς)

ΔΙΑΣΧΙΣΗ

Δήλωση συναρτήσεων (γραμμές 5-17)

Βλ. ενότητα «Συναρτήσεις», σελίδες «13-14».

Όπου "ω" η γωνία σε ακτίνια και "Ω" η γωνία σε μοίρες (γραμμή 18)

Στη γραμμή «18» αναφέρεται μία επισήμανση για τα ορίσματα των συναρτήσεων που είναι καταγεγραμμένα στα σχόλια του πηγαίου κώδικα για χάριν ευκολίας. Για παράδειγμα, η συνάρτηση «Sin (rad_S)» που υλοποιείται στις γραμμές «77-84», δίπλα στο σχόλιο αναφέρεται ως «Sin (ω)», όπου το «ω» χαρακτηρίζει τη μεταβλητή «rad_S». Αντίστοιχα και στη συνάρτηση «Deg_to_Rad (deg_DtR)» που υλοποείται στις γραμμές «58-65», δίπλα στο σχόλιο αναφέρεται ως «Deg_to_Rad (Ω)», όπου το «Ω» χαρακτηρίζει τη μεταβλητή «deg_DtR». Η διάσχιση των συναρτήσεων έχεις ως εξής :

main (int argc, char **argv) (γραμμές 20-37)

Στις γραμμές «20-37» υλοποιείται η κύρια συνάρτηση «main(…)», τύπου «int», του προγράμματος που επιστρέφει τον ακέραιο αριθμό «0», όταν το πρόγραμμα τρέχει χωρίς κανένα πρόβλημα. Η διάσχιση της «main(...)» έχει ως

εξής :

Δήλωση μεταβλητών (γραμμή 26)

Βλ. ενότητα «Μεταβλητές», υποενότητα «main (int argc, char **argv)», σελίδα «14».

Κλήση της συνάρτησης "Title ()" (γραμμή 26)

Στη γραμμή «26», η «main(...)» καλεί τη συνάρτηση «Title(...)» που υλοποιείται στις γραμμές «39-44».

Κλήση της συνάρτησης "Read_Deg ()" (γραμμή 27)

Στη γραμμή «27», η «main(...)» καλεί τη συνάρτηση «Read_Deg(...)» που υλοποιείται στις γραμμές «46-56» και η πραγματική τιμή που επιστρέφει η δεύτερη, καταχωρείται στην πραγματική μεταβλητή «deg».

Κλήση της συνάρτησης "Deg_to_Rad (Ω)" (γραμμή 28)

Στη γραμμή «28», η «main(…)» καλεί τη συνάρτηση «Deg_to_Rad(…)» που υλοποιείται στις γραμμές «58-65» και η πραγματική τιμή που επιστρέφει η δεύτερη, καταχωρείται στην πραγματική μεταβλητή «rad».

Κλήση της συνάρτησης "Print_Deg (Ω)" (γραμμή 29)

Στη γραμμή «29», η «main(...)» καλεί τη συνάρτηση «Print_Deg(...)» που υλοποιείται στις γραμμές «67-70».

Κλήση της συνάρτησης "Print_Rad (ω)" (γραμμή 30)

Στη γραμμή «30», η «main(...)» καλεί τη συνάρτηση «Print_Rad(...)» που υλοποιείται στις γραμμές «72-75».

Κλήση της συνάρτησης "Print_Sin_TaylorS (ω)" (γραμμή 31)

Στη γραμμή «31», η «main(...)» καλεί τη συνάρτηση «Print_Sin_TaylorS(...)» που υλοποιείται στις γραμμές «122-130».

Κλήση της συνάρτησης "Check_Sin_TaylorS (ω)" (γραμμή 32)

Στη γραμμή «32», η «main(...)» καλεί τη συνάρτηση «Check_Sin_TaylorS (...)» που υλοποιείται στις γραμμές «132-151».

Κλήση της συνάρτησης "Print_Cos_Taylor_S (ω)" (γραμμή 33)

Στη γραμμή «33», η «main(...)» καλεί τη συνάρτηση «Print_Cos_TaylorC(...)» που υλοποιείται στις γραμμές «198-206».

Κλήση της συνάρτησης "Check_Cos_TaylorS (ω)" (γραμμή 34)

Στη γραμμή «34», η «main(...)» καλεί τη συνάρτηση «Check_Cos_TaylorC (...)» που υλοποιείται στις γραμμές «208-227».

Title () (γραμμές 39-44)

Στις γραμμές «39-44» υλοποιείται η συνάρτηση «Title(…), τύπου «void», που τυπώνει τον τίτλο του προγράμματος. Η διάσχιση της «Title(…)» έχει ώς εξής :

Τίτλος προγράμματος (γραμμή 42)

Στη γραμμή «42» τυπώνεται από τη «standard» έξοδο με μία συνάρτηση «printf(…)» το μήνυμα «Υπολογισμός του ημιτόνου και του συνημιτόνου μίας γωνίας» με δύο χαρακτήρες διαφυγής της αλλαγής γραμμής (\n), που απεικονίζουν τον τίτλο του προγράμματος.

Read_Deg () (γραμμές 46-56)

Στις γραμμές «46-56» υλοποιείται η συνάρτηση «Read_Deg(…)», τύπου «double», που διαβάζει τη γωνία από τη «standard» είσοδο σε μοίρες και την επιστρέφει εκεί που είναι ο έλεγχος του προγράμματος. Η διάσχιση της «Read_Deg(…)» έχει ως εξής:

Δήλωση μεταβλητών (γραμμή 48)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Read_Deg ()», σελίδα «14».

Εισαγωγή της γωνίας σε μοίρες (γραμμή 52)

Στη γραμμή «52» διαβάζεται από τη «standard» είσοδο με μία συνάρτηση «scanf(…)» μία γωνία σε μοίρες συνοδευόμενο με το κατάλληλο μήνυμα που τυπώνεται από τη «standard» έξοδο με μία συνάρτηση «printf(…)» στη γραμμή «51» και καταχωρείται στη μεταβλητή «deg_RD». Η γωνία αυτή συνιστάται να ανήκει στο διάστημα του πρώτου κύκλου [0,360] για να μην δημιουργηθούν προβλήματα ως προς την ορθότητα των αποτελεσμάτων. Βέβαια, επισημαίνεται με το χαρακτηριστικό μήνυμα που τυπώνεται από τη «standard» έξοδο με μία συνάρτηση «printf(…)» στη γραμμή «50», χωρίς ωστόσο, να απαγορεύεται.

Επιστροφή της γωνίας σε μοίρες (γραμμή 55)

Στη γραμμή «55» η εντολή «return deg_RD» επιστρέφει το περιεχόμενο της μεταβλητής «deg_RD», δηλαδή, της γωνίας που διαβάστηκε από τη «standard» είσοδο σε μοίρες, εκεί που είναι ο έλεγχος του προγράμματος.

Deg_to_Rad (Ω) (γραμμές 58-65)

Στις γραμμές «58-65» υλοποιείται η συνάρτηση «Deg_to_Rad(...)», τύπου «double» και με παράμετρο τη μεταβλητή «deg_DtR» (τύπου «double»), όπου, παίρνει σαν είσοδο τη γωνία που εισήχθη σε μοίρες, την μετατρέπει σε ακτίνια και επιστρέφει το αποτέλεσμα εκεί που είναι ο έλεγχος. Η διάσχιση της «Deg_to_Rad(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 60)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Deg_to_Rad (deg_DtR)», σελίδα «14».

Μετατροπή της γωνίας από μοίρες σε ακτίνια (γραμμή 62)

Στη γραμμή «62» υλοποιείται ο μαθηματικός υπολογισμός «(pi * deg_DtR) / 180», της μετατροπής της γωνίας από μοίρες σε ακτίνια και το αποτέλεσμα καταχωρείται στη μεταβλητή «rad_dtr».

Επιστροφή της γωνίας σε ακτίνια (γραμμή 64)

Στη γραμμή «64» η εντολή «return rad_dtr» επιστρέφει το περιεχόμενο της μεταβλητής «rad_dtr», δηλαδή, τη γωνία που εισήχθη από τη «standard» είσοδο σε ακτίνια εκεί που είναι ο έλεγχος του προγράμματος.

Print_Deg (Ω) (γραμμές 67-70)

Στις γραμμές «67-70» υλοποιείται η συνάρτηση «Print_Deg(…)», τύπου «void» και με παράμετρο τη μεταβλητή «deg_PD» (τύπου «double»), παίρνει σαν είσοδο τη γωνία που εισήχθη από τη «standard» είσοδο και τυπώνει από τη «standard» έξοδο το περιεχόμενο της μεταβλητής που έχει καταχωρηθεί. Η διάσχιση της «Print_Deg(…)» έχει ως εξής :

Εκτύπωση της γωνίας σε μοίρες (γραμμή 69)

Στη γραμμή «69» τυπώνεται από τη «standard» έξοδο το περιεχόμενο της μεταβλητής «deg_PD» (η γωνία σε μοίρες που επιστρέφει η συνάρτηση «Read_Deg(…)») με μία συνάρτηση «printf()» συνοδευόμενο με το κατάλληλο μήνυμα. Αξίζει να σημειωθεί ότι το αλφαριθμητικό μορφοποίησης είναι το «%20.6lf», δηλαδή, το αποτέλεσμα θα τυπωθεί αμεσώς μετά από «20» κενούς χαρακτήρες με «6» δεκαδικά ψηφία, καθώς, πρόκειται για πραγματικό διπλής ακρίβειας. Αυτό αποσκοπεί στην ομοιόμορφη στοίχιση των αποτελεσμάτων.

Print_Rad (ω) **(γραμμές 72-75)**

Στις γραμμές «72-75» υλοποιείται η συνάρτηση «Print_Rad(…)», τύπου «void» και με παράμετρο τη μεταβλητή «rad_PR» (τύπου «double»), παίρνει σαν είσοδο τη γωνία που εισήχθη, σε ακτίνια και τυπώνει από τη «standard» έξοδο το περιεχόμενο της μεταβλητής που έχει καταχωρηθεί. Η διάσχιση της «Print_Rad(…)» έχει ως εξής :

Εκτύπωση της γωνίας σε ακτίνια (γραμμή 74)

Στη γραμμή «74» τυπώνεται από τη «standard» έξοδο το περιεχόμενο της μεταβλητής «rad_PR» (η γωνία σε ακτίνια που επιστρέφει η συνάρτηση «Deg_to_Rad(…)») με μία συνάρτηση «printf()» συνοδευόμενο με το κατάλληλο μήνυμα. Αξίζει να σημειωθεί ότι το αλφαριθμητικό μορφοποίησης είναι το «%20.6lf», δηλαδή, το αποτέλεσμα θα τυπωθεί αμεσώς μετά από «20» κενούς χαρακτήρες με «6» δεκαδικά ψηφία, καθώς, πρόκειται για πραγματικό διπλής ακρίβειας. Αυτό αποσκοπεί στην ομοιόμορφη στοίχιση των αποτελεσμάτων.

Sin (ω) **(γραμμές 77-84)**

Στις γραμμές «77-84» υλοποιείται η συνάρτηση «Sin(…)», τύπου «double» και με παράμετρο τη μεταβλητή «rad_S» (τύπου «double»), όπου, παίρνει σαν είσοδο τη γωνία που εισήχθη, σε ακτίνια, υπολογίζει το ημιτόνο της με την έτοιμη συνάρτηση «sin(…)» που εισάγεται με τη βιβλιοθήκη «math.h» και επιστρέφει το αποτέλεσμα εκεί που είναι ο έλεγχος του προγράματος. Η διάσχιση της «Sin(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 79)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Sin (rad_S)», σελίδα «15».

Υπολογισμός του ημιτόνου της γωνίας με τη συνάρτηση "sin (ω)" (γραμμή 81)

Στη γραμμή «81» υπολογίζεται με την έτοιμη συνάρτηση «sin(…)» που εισάγεται με τη βιβλιοθήκη «math.h», το ημίτονο της γωνίας που εισήχθη, σε ακτίνια και το αποτέλεσμα καταχωρείται στη μεταβλητή «c».

Επιστροφή του ημιτόνου της γωνίας με τη συνάρτηση "sin (ω)" (γραμμή 83)

Στη γραμμή «83» η εντολή «return c» επιστρέφει το περιεχόμενο της μεταβλητής «c», δηλαδή, το ημίτονο της γωνίας που εισήχθη, σε ακτίνια με την έτοιμη συνάρτηση «sin(…)» που εισάγεται με τη βιβλιοθήκη «math.h», εκεί που είναι ο έλεγχος του προγράμματος.

Taylor_S (ω) (γραμμές 86-120)

Στις γραμμές «86-120» υλοποιείται η συνάρτηση «Taylor_S(…)», τύπου «double» και με παράμετρο τη μεταβλητή «rad_TaylorS» (τύπου «double»), όπου, παίρνει για είσοδο τη γωνία που εισήχθη, σε ακτίνια, υπολογίζει το ημίτονο της με την χαρακτηριστική απειροσειρά «Taylor» και επιστρέφει το αποτέλεσμα εκεί που είναι ο έλεγχος του προγράμματος. Η διάσχιση της «Taylor_S(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμές 88-92)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Taylor_S (rad_TaylorS)», σελίδες «15-16».

Αρχικοποίηση μεταβλητών (γραμμές 89, 91, 92)

Στις προαναφερόμενες γραμμές γίνονται μερικές αρχικοποιήσεις μεταβλητών. Πιο αναλυτικά, στη γραμμή «89» η τιμή «0.0» εκχωρείται στη μεταβλητή «sin_T» για το άθροισμα του ημιτόνου της γωνίας σε ακτίνια με την απειροσειρά «Taylor», στη γραμμή «92» η τιμή «1» στη «j» ως βοηθητική μεταβλητή για τον υπολογισμό του δευτέρου όρου, του τρίτου κ.ο.κ. και στη γραμμή «91» η τιμή «-1» στη «sign» ως βοηθητική μεταβλητή για την αλλαγή του προσήμου στην πρόσθεση των όρων της απειροσειράς «Taylor».

1ος Βρόχος (γραμμές 94-117)

Στις γραμμές «94-117» υλοποιείται ένας βρόχος με την εντολή «do – while» με συνθήκη τερματισμού τη παράσταση «abs_diff_terms > 0.000001».

Πιο αναλυτικά, ο πρώτος βρόχος εκτελείται τουλάχιστον μία φορά και εφόσον, η παράσταση «abs_diff_terms > 0.000001» παράγει αποτέλεσμα μία τιμή «True» (συνθήκη αληθής) θα συνεχίσει να εκτελείται εώς ότου παράγει αποτέλεσμα τη τιμή «False» (συνθήκη ψευδής). Η διάσχιση του «1^{ου} βρόχου» έχει ώς εξής :

Αρχικοποίηση μεταβλητής του πρώτου όρου, του δευτέρου ... (γραμμή 96)

Στη γραμμή «96» γίνεται μια αρχικοποίηση της μεταβλητής «term» που χαρακτηρίζει τον πρώτο όρο, τον δεύτερο κ.ο.κ της απειροσειράς, η οποία θα γίνεται κάθε φορά που εκτελείται ο «1^{ος} βρόχος». Πιο αναλυτικά, κάθε φορά που θα εκτελείται ο βρόχος στη «term» θα καταχωρείται η τιμή «1».

2^{ος} Βρόχος (γραμμές 97-100)

Στις γραμμές «97-100» υλοποιείται ένας άλλος βρόχος μεσα στον πρώτο, με την εντολή «for» και με συνθήκη τερματισμού τη συνθήκη «i <= j», δηλαδή, για όσο η βοηθητική μεταβλητή ελέγχου του «2^{ου} βρόχου» είναι μικρότερη ή ίση της βοηθητικής μεταβλητής για τον υπολογισμό του δεύτερου όρου, του τρίτου

κ.ο.κ. της απειροσειράς. Η βοηθητική «i» έχει για αρχική τιμή, την τιμή «1» και κάθε φορά που εκτελείται ο «2ος βρόχος» η τιμή της αυξάνεται κατά ένα με την παράσταση «i++». Η διάσχιση του «2ου βρόχου» έχει ως εξής :

Υπολογισμός του πρώτου όρου, του δευτέρου ... (γραμμή 99)

Στη γραμμή «99» υλοποιείται η παράσταση «term * (rad_TaylosS / i)» για τον υπολογισμό του πρώτου όρου, του δευτέρου κ.ο.κ. (ω^1 / 1!, ω^3 / 3! ...) της απειροσειράς. Το αποτέλεσμα καταχωρείται στη μεταβλητή «term», κάθε φορά που εκτελείται ο «1ος βρόχος» και ο «2ος βρόχος». Η «term» σε κάθε εκτέλεση του πρώτου βρόχου έχει για αρχική τιμή, τη τιμή «1» (βλ. «Αρχικοποίηση μεταβλητής του πρώτου όρου, του δευτέρου ... (γραμμή 96)»).

Αύξηση της βοηθητικής μεταβλητής για τον υπολογισμό του δευτέρου όρου, του τρίτου ... (γραμμή 101)

Στη γραμμή «101» η βοηθητική μεταβλητή «j» για τον υπολογισμό του δευτέρου όρου, του τρίτου κ.ο.κ. της απειροσειράς, αυξάνεται κατά «2» κάθε φορά που εκτελείται ο «1ος βρόχος».

Υπολογισμός του δεύτερου όρου, του τρίτου ... (γραμμή 102)

Στη γραμμή «102» υλοποιείται η παράσταση «term * ((rad_TaylorS * rad_TaylorS) / (j * (j – 1)))» για τον υπολογισμό του δεύτερου όρου, του τρίτου κ.ο.κ. (ω^3 / 3!, ω^5 / 5! ...) της απειροσειράς. Το αποτέλεσμα καταχωρείται στη μεταβλητή «next_term», κάθε φορά που εκτελείται ο «1ος βρόχος»

Υπολογισμός της διαφοράς του δεύτερου όρου με τον πρώτο, του τρίτου με τον δεύτερο ... (γραμμή 103)

Στη γραμμή «103» καταχωρείται στη μεταβλητή «diff_terms» η διαφορά του δεύτερου όρου με τον πρώτο, του τρίτου με τον δεύτερο κ.ο.κ. της απειροσειράς, κάθε φορα που εκτελείται ο «1ος βρόχος».

Υπολογισμός της απόλυτης τιμής της διαφοράς των όρων (γραμμή 104)

Στη γραμμή «104» καταχωρείται στη μεταβλητή «abs_diff_terms» η απόλυτη τιμή της διαφοράς του δεύτερου όρου με τον πρώτο, του τρίτου με τον δεύτερο κ.ο.κ της απειροσειράς, κάθε φορά που εκτελείται ο «1ος βρόχος». Η απόλυτη τιμή υπολογίζεται με τη βοήθεια της συνάρτησης «fabs(…)» που εισάγεται με τη βιβλιοθήκη «math.h».

(~) 1η επανάληψη του 1ου βρόχου (γραμμές 105-110)

(~) 2η, 3η ... επανάληψη του 1ου βρόχου (γραμμές 111-115)

Στις γραμμές «105-115» υλοποιείται ένας έλεγχος με την εντολή «if – else»

για τον αριθμό επανάληψης του «1ου βρόχου» που βρίσκεται το πρόγραμμα. Η συνθήκη ελέγχου είναι η παράσταση «j == 3» που ελέγχει το περιεχόμενο της βοηθητικής μεταβλητής «j» για τον υπολογισμό του δευτέρου όρου, του τρίτου κ.ο.κ. της απειροσειράς. Η συνθήκη ελέγχου πραγματοποιείται, διότι, οι χαρακτηριστικές πράξεις για τον υπολογισμό του ημιτόνου μιας γωνίας σε ακτίνια δεν διαφέρουν μεταξύ τους από το δεύτερο άθροισμα και μετά. Προφανώς, το άθροισμα του πρώτου με τον δεύτερο όρο είναι αυτό που διαφέρει ελάχιστα από τα άλλα αθροίσματα.

(~) 1η επανάληψη του 1ου βρόχου (γραμμές 105-110)

Στις γραμμές «107-112» εκτελούνται οι εντολές της «if (j == 3)» (γραμμή 105) για την περίπτωση που η «j» περιέχει την τιμή «3», δηλαδή, ότι το πρόγραμμα βρίσκεται στην πρώτη εκτέλεση του «1ου βρόχου». Συνεπώς, υπολογίζεται το πρώτο άθροισμα της απειροσειράς που διαφέρει ελάχιστα από τα υπόλοιπα. Για «ω» γωνία σε ακτίνια υπολογίζεται η παράσταση «ω^1 / 1! - ω^3 / 3!».

Υπολογισμός του πρώτου αθροίσματος με το κατάλληλο πρόσημο "sign" (γραμμή 107)

Στη γραμμή «107» υλοποιείται η παράσταση «term + (sign * next_term)» για το πρώτο άθροισμα (του πρώτου όρου με τον δεύτερο) με το κατάλληλο πρόσημο που καθορίζει η μεταβλητή «sign» που έχει για αρχική τιμή, τη τιμή «-1» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 91)». Το αποτέλεσμα καταχωρείται στη μεταβλητή «first_sin_T».

Καταχώρηση του αθροίσματος στη μεταβλητή "sin_T" (γραμμή 108)

Στη γραμμή «108» καταχωρείται με την παράσταση «sin_T + first_sin_T» το πρώτο άθροισμα με το κατάλληλο πρόσημο στη μεταβλητή «sin_T» που έχει για αρχική τιμή την τιμή «0.0» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 89)» .

Αλλαγή προσήμου (γραμμή 109)

Στη γραμμή «109» καταχωρείται στη μεταβλητή «sign» η χαρακτηριστική παράσταση «sign * (-1)» που βοηθάει στο να αλλάζει το πρόσημο της απειροσειράς αμέσως μετά το πρώτο άθροισμα και μετά.

(~) 2η, 3η ... επανάληψη του 1ου βρόχου (γραμμές 111-115)

Στις γραμμές «111-115» εκτελούνται οι εντολές της «else» (γραμμή 111) που αντιστοιχεί στην «if (j == 3)» (γραμμή 105), για την περίπτωση που η «j» δεν περιέχει την τιμή «3», δηλαδή, ότι το πρόγραμμα βρίσκεται στην δεύτερη εκτέλεση του «1ου βρόχου» και μετά. Συνεπώς υπολογίζεται το δεύτερο άθροισμα, το τρίτο κ.ο.κ. της απειροσειράς. Για «ω» γωνία σε ακτίνια υπολογίζεται η παράσταση «(ω^1 / 1! - ω^3 / 3!) + ω^5 / 5! ...».

Υπολογισμός του δευτέρου αθροίσματος, του τρίτου ... με το κατάλληλο πρόσημο "sign" (γραμμή 113)

Στη γραμμή «113» υλοποιείται η παράσταση «(sin_T + (sign * next_term)» για το δεύτερο άθροισμα, το τρίτο κ.ο.κ. της απειροσειράς με το κατάλληλο πρόσημο που καθορίζει η μεταβλητή «sign», όπου η «sign» έχει περιεχόμενο τη τιμή «1» μετά την πρώτη εκτέλεση του «1ου βρόχου». Το αποτέλεσμα καταχωρείται στη μεταβλητή «sin_T», όπου η «sin_T» έχει περιεχόμενο το πρώτο άθροισμα μετά την πρώτη εκτέλεση του «1ου βρόχου».

Αλλαγή προσήμου (γραμμή 114)

Στη γραμμή «114» καταχωρείται στη μεταβλητή «sign» η χαρακτηριστική παράσταση «sign * (-1)» που βοηθάει στο να αλλάζει το πρόσημο της απειροσειράς αμέσως μετά το δεύτερο άθροισμα και μετά.

Επιστροφή του αθροίσματος των όρων (απειροσειρά Taylor) (γραμμή 119)

Στη γραμμή «119» η εντολή «return sin_T» επιστρέφει το συνολικό άθροισμα των όρων της απειροσειράς, εκεί που είναι ο έλεγχος του προγράμματος.

Print_Sin_TaylorS (ω) (γραμμές 122-130)

Στις γραμμές «122-130» υλοποιείται η συνάρτηση «Print_Sin_TaylorS(…)», τύπου «void» και με παράμετρο τη μεταβλητή «rad_PSTS» (τύπου «double»), όπου δέχεται για είσοδο τη γωνία που εισήχθη, σε ακτίνια, καλεί τις συναρτήσεις «Sin(…)», «Taylor_S(…) και τυπώνει τις τιμές που επιστρέφουν, δηλαδή, το ημίτονο της γωνίας με την έτοιμη συνάρτηση «sin(...)» και το ημίτονο με την απειροσειρά «Taylor» αντίστοιχα. Η διάσχιση της «Print_Sin_TaylorS(…) έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 124)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Print_Sin_TaylorS (rad_PSTS)» σελίδα «16»

Κλήση της συνάρτησης "Sin (ω)" (γραμμή 126)

Στη γραμμή «126» η «Print_Sin_TaylorS(…)» καλεί τη συνάρτηση «Sin(…)» που επιστρέφει το ημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση «sin(...)». Το αποτέλεσμα καταχωρείται στη μεταβλητή «rad_Sin».

Εκτύπωση του ημιτόνου της γωνίας με την έτοιμη συνάρτηση "sin (ω)" (γραμμή 127)

Στη γραμμή «127» τυπώνεται από τη «standard» έξοδο το περιεχόμενο της μεταβλητής «rad_Sin» (το ημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση «sin()», που επιστρέφει η συνάρτηση «Sin(…)») με μία συνάρτηση «printf()» συνοδευόμενο με το κατάλληλο μήνυμα. Αξίζει να σημειωθεί ότι το αλφαριθμητικό μορφοποίησης είναι το «%20.6lf», δηλαδή, το αποτέλεσμα θα τυπωθεί αμεσώς μετά από «20» κενούς χαρακτήρες με «6» δεκαδικά ψηφία, καθώς, πρόκειται για πραγματικό διπλής ακρίβειας. Αυτό αποσκοπεί στην ομοιόμορφη στοίχιση των αποτελεσμάτων.

Κλήση της συνάρτησης "Taylor_S (ω)" (γραμμή 128)

Στη γραμμή «128» η «Print_Sin_TaylorS(…)» καλεί τη συνάρτηση «Taylor_S(…)» που επιστρέφει το ημίτονο της γωνίας σε ακτίνια, με τη χαρακτηριστική απειροσειρά «Taylor». Το αποτέλεσμα καταχωρείται στη μεταβλητή «rad_TaylorSin».

Εκτύπωση του ημιτόνου της γωνίας με την απειροσειρά "Taylor" (γραμμή 129)

Στη γραμμή «129» τυπώνεται από τη «standard» έξοδο το περιεχόμενο της μεταβλητής «rad_TaylorSin» (το ημίτονο της γωνίας σε ακτίνια με την απειροσειρά «Taylor» που επιστρέφει η συνάρτηση «Taylor_S(…)») με μία συνάρτηση «printf()» συνοδευόμενο με το κατάλληλο μήνυμα. Αξίζει να σημειωθεί ότι το αλφαριθμητικό μορφοποίησης είναι το «%20.6lf», δηλαδή, το αποτέλεσμα θα τυπωθεί αμεσώς μετά από «20» κενούς χαρακτήρες με «6» δεκαδικά ψηφία, καθώς, πρόκειται για πραγματικό διπλής ακρίβειας. Αυτό αποσκοπεί στην ομοιόμορφη στοίχιση των αποτελεσμάτων.

Check_Sin_TaylorS (ω) (γραμμές 132-149)

Στις γραμμές «132-149» υλοποιείται η συνάρτηση «Check_Sin_Taylor_S (...)», τύπου «void» και με παράμετρο τη μεταβλητή «rad_CSTS», όπου δέχεται για είσοδο τη γωνία που εισήχθη, σε ακτίνια, καλεί τις συναρτήσεις «Sin(…)», «Taylor_S(…) και τυπώνει τις τιμές που επιστρέφουν, δηλαδή, το ημίτονο της γωνίας με την έτοιμη συνάρτηση «sin(...)» και το ημίτονο με την απειροσειρά «Taylor» αντίστοιχα. Υπολογίζει την απόλυτη τιμή της διαφοράς των δύο αριθμών που υπολογίζουν το ημίτονο και συγκρίνει αν είναι «σχεδόν» ίσοι ή όχι. Η διάσχιση της «Check_Sin_TaylorS(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμές 134-135)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Check_Sin_TaylorS (rad_CSTS) σελίδα «16»

Κλήση της συνάρτησης "Sin (ω)" (γραμμή 137)

Στη γραμμή «137» η «Check_Sin_TaylorS(…)» καλεί τη συνάρτηση «Sin(…)» που επιστρέφει το ημίτονο της γωνίας σε ακτίνια με την έτοιμη

συνάρτηση «sin(...)». Το αποτέλεσμα καταχωρείται στη μεταβλητή «CheckSin».

Κλήση της συνάρτησης "Taylor_S (ω)" (γραμμή 138)

Στη γραμμή «138» η «Check_Sin_TaylorS(…)» καλεί τη συνάρτηση «Taylor_S(…)» που επιστρέφει το ημίτονο της γωνίας σε ακτίνια, με τη χαρα - κτηριστική απειροσειρά «Taylor». Το αποτέλεσμα καταχωρείται στη μεταβλητή «CheckTaylorS».

Υπολογισμός της διαφοράς της συνάρτησης "sin (ω)" με την απειροσειρά "Taylor" (γραμμή 139)

Στη γραμμή «139» υλοποιείται η παράσταση «CheckSin – CheckTaylorS» για τον υπολογισμό της διαφοράς του ημιτόνου της γωνίας σε ακτίνιας με την έτοιμη συνάρτηση «sin()» και την απειροσειρά «Taylor». Το αποτέλεσμα καταχωρείται στη μεταβλητή «diff_CheckSinTaylorS»

Υπολογισμός της απόλυτης τιμής της διαφοράς της συνάρτησης "sin (ω)" με την απειροσειρά "Taylor" (γραμμή 140)

Στη γραμμή «142» υπολογίζεται η απόλυτη τιμή της διαφοράς των δύο αριθμών που βγάζουν το ημίτονο της γωνίας. Η απόλυτη τιμή υπολογίζεται με τη βοήθεια της συνάρτησης «fabs(…)» που εισάγεται με τη βιβλιοθήκη «math.h».Το αποτέλεσμα καταχωρείται στη μεταβλητή «abs_diff_CheckSin TaylorS».

(~) Αποδεκτή απόλυτη τιμή της διαφοράς της διαφοράς του ημιτόνου με την απειροσειρά "Taylor" (γραμμές 141-145)
(~) Απορριπτέα απόλυτη τιμή της διαφοράς της διαφοράς του ημιτόνου με την απειροσειρά "Taylor" (γραμμές 146-150)

Στις γραμμές «143-150» υλοποιείται ένας έλεγχος με την εντολή «if – else» ως προς την απόκλιση της απόλυτης τιμής της διαφοράς των δύο αριθμών που βγάζουν το ημίτονο της γωνίας. Η συνθήκη ελέγχου είναι η παράσταση «abs_diff_ CheckSinTaylorS <= 0.0000009». Ο έλεγχος πραγματοποιείται για να διαπιστωθεί αν οι δύο αριθμοί είναι «σχεδόν» ίσοι.

(~) Αποδεκτή απόλυτη τιμή της διαφοράς της συνάρτησης "sin (ω)" με την απειροσειρά "Taylor" (γραμμές 141-145)

Στις γραμμές «121-124» εκτελούνται οι εντολές της «if (abs_diff_ CheckSinTaylorS)» (γραμμή 121) για την περίπτωση που η απόλυτη τιμή της διαφοράς των δύο αριθμών είναι μικρότερη ή ίση από την τιμή «0.0000009» και επομένως, είναι «σχεδόν» ίσοι. Στις γραμμές «143-144» τυπώνονται στη «standard» έξοδο με τη συνάρτηση «printf(…)» τα χαρακτηριστικά μηνύματα.

(~) Απορριπτέα απόλυτη τιμή της διαφοράς της συνάρτησης "sin (ω)" με την απειροσειρά "Taylor" (γραμμές 146-150)

Στις γραμμές «146-150» εκτελούνται οι εντολές της «else» που αντιστοιχεί στην «if (abs_diff_ CheckSinTaylorS)» (γραμμή 141) για την περίπτωση που η απόλυτη τιμή της διαφοράς των δύο αριθμών είναι μεγαλύτερη από την τιμή «0.0000009» και επομένως, οι δύο αριθμοί δεν είναι «σχεδόν» ίσοι. Στις γραμμές «148-149» τυπώνονται στη «standard» έξοδο με τη συνάρτηση «printf(…)» τα χαρακτηριστικά μηνύματα.

Cos (ω) (γραμμές 153-160)

Στις γραμμές «153-160» υλοποιείται η συνάρτηση «Cos(…)», τύπου «double» και με παράμετρο τη μεταβλητή «rad_C» (τύπου «double»), όπου, παίρνει σαν είσοδο τη γωνία που εισήχθη, σε ακτίνια, υπολογίζει το συνημιτόνο της με την έτοιμη συνάρτηση «cos(…)» που εισάγεται με τη βιβλιοθήκη «math.h» και επιστρέφει το αποτέλεσμα εκεί που είναι ο έλεγχος του προγράματος. Η διάσχιση της «Cos(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 155)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Cos (rad_C)» σελίδες «16-17».

Υπολογισμός του συνημιτόνου της γωνίας με τη συνάρτηση "cos (ω)" (γραμμή 157)

Στη γραμμή «157» υπολογίζεται με την έτοιμη συνάρτηση «cos(…)» που εισάγεται με τη βιβλιοθήκη «math.h», το συνημίτονο της γωνίας που εισήχθη, σε ακτίνια και το αποτέλεσμα καταχωρείται στη μεταβλητή «d».

Επιστροφή του συνημιτόνου της γωνίας με τη συνάρτηση "cos (ω)" (γραμμή 159)

Στη γραμμή «159» η εντολή «return d» επιστρέφει το περιεχόμενο της μεταβλητής «d», δηλαδή, το συνημίτονο της γωνίας που εισήχθη, σε ακτίνια με την έτοιμη συνάρτηση «cos(…)» που εισάγεται με τη βιβλιοθήκη «math.h», εκεί που είναι ο έλεγχος του προγράμματος.

Taylor_C (ω) (γραμμές 162-196)

Στις γραμμές «162-196» υλοποιείται η συνάρτηση «Taylor_C(…)», τύπου «double» και με παράμετρο τη μεταβλητή «rad_TaylorC» (τύπου «double»), όπου, παίρνει για είσοδο τη γωνία που εισήχθη, σε ακτίνια, υπολογίζει το συνημίτονο της με την χαρακτηριστική απειροσειρά «Taylor» και επιστρέφει το αποτέλεσμα εκεί που είναι ο έλεγχος του προγράμματος. Η διάσχιση της «Taylor_C(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμές 164-168)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Taylor_C (rad_TaylorC)», σελίδα «17».

Αρχικοποίηση μεταβλητών (γραμμές 165, 167, 168)

Στις προαναφερόμενες γραμμές γίνονται μερικές αρχικοποιήσεις μεταβλητών. Πιο αναλυτικά, στη γραμμή «165» η τιμή «0.0» εκχωρείται στη μεταβλητή «cos_T» για το άθροισμα του συνημιτόνου της γωνίας σε ακτίνια με την απειροσειρά «Taylor», στη γραμμή «168» η τιμή «0» στη «j» ως βοηθητική μεταβλητή για τον υπολογισμό του δευτέρου όρου, του τρίτου κ.ο.κ. και στη γραμμή «167» η τιμή «-1» στη «sign» ως βοηθητική μεταβλητή για την αλλαγή του προσήμου στην πρόσθεση των όρων της απειροσειράς «Taylor».

1ος Βρόχος (γραμμές 170-193)

Στις γραμμές «170-193» υλοποιείται ένας βρόχος με την εντολή «do – while» με συνθήκη τερματισμού τη παράσταση «abs_diff_terms > 0.000001».

Πιο αναλυτικά, ο πρώτος βρόχος εκτελείται τουλάχιστον μία φορά και εφόσον, η παράσταση «abs_diff_terms > 0.000001» παράγει αποτέλεσμα μία τιμή «True» (συνθήκη αληθής) θα συνεχίσει να εκτελείται εώς ότου παράγει αποτέλεσμα τη τιμή «False» (συνθήκη ψευδής). Η διάσχιση του «1^{ου} βρόχου» έχει ώς εξής :

Αρχικοποίηση μεταβλητής του πρώτου όρου, του δευτέρου ... (γραμμή 172)

Στη γραμμή «172» γίνεται μια αρχικοποίηση της μεταβλητής «term» που χαρακτηρίζει τον πρώτο όρο, τον δεύτερο κ.ο.κ της απειροσειράς, η οποία θα γίνεται κάθε φορά που εκτελείται ο «1^{ος} βρόχος». Πιο αναλυτικά, κάθε φορά που θα εκτελείται ο βρόχος στη «term» θα καταχωρείται η τιμή «1».

2^{ος} Βρόχος (γραμμές 173-176)

Στις γραμμές «173-176» υλοποιείται ένας άλλος βρόχος μεσα στον πρώτο, με την εντολή «for» και με συνθήκη τερματισμού τη συνθήκη «i <= j», δηλαδή, για όσο η βοηθητική μεταβλητή ελέγχου του «2^{ου} βρόχου» είναι μικρότερη ή ίση της βοηθητικής μεταβλητής για τον υπολογισμό του δεύτερου όρου, του τρίτου κ.ο.κ. της απειροσειράς. Η βοηθητική «i» έχει για αρχική τιμή, την τιμή «1» και κάθε φορά που εκτελείται ο «2^{ος} βρόχος» η τιμή της αυξάνεται κατά ένα με την παράσταση «i++». Η διάσχιση του «2^{ου} βρόχου» έχει ως εξής :

Υπολογισμός του πρώτου όρου, του δευτέρου ... (γραμμή 175)

Στη γραμμή «99» υλοποιείται η παράσταση «term * (rad_TaylosC / i)» για τον υπολογισμό του πρώτου όρου, του δευτέρου κ.ο.κ. (1, ω^2 / 2! ...) της απειροσειράς. Το αποτέλεσμα καταχωρείται στη μεταβλητή «term», κάθε φορά που εκτελείται ο «1^{ος} βρόχος» και ο «2^{ος} βρόχος». Η «term» σε κάθε εκτέλεση

του πρώτου βρόχου έχει για αρχική τιμή, τη τιμή «1» (βλ. «Αρχικοποίηση μεταβλητής του πρώτου όρου, του δευτέρου ... (γραμμή 172)»).

Αύξηση της βοηθητικής μεταβλητής για τον υπολογισμό του δευτέρου όρου, του τρίτου ... (γραμμή 177)

Στη γραμμή «177» η βοηθητική μεταβλητή «j» για τον υπολογισμό του δευτέρου όρου, του τρίτου κ.ο.κ. της απειροσειράς, αυξάνεται κατά «2» κάθε φορά που εκτελείται ο «1^{ος} βρόχος».

Υπολογισμός του δεύτερου όρου, του τρίτου ... (γραμμή 178)

Στη γραμμή «178» υλοποιείται η παράσταση «term * ((rad_TaylorC * rad_TaylorC) / (j * (j – 1)))» για τον υπολογισμό του δεύτερου όρου, του τρίτου κ.ο.κ. (ω^2 / 2!, ω^4 / 4! ...) της απειροσειράς. Το αποτέλεσμα καταχωρείται στη μεταβλητή «next_term», κάθε φορά που εκτελείται ο «1^{ος} βρόχος»

Υπολογισμός της διαφοράς του δεύτερου όρου με τον πρώτο, του τρίτου με τον δεύτερο ... (γραμμή 179)

Στη γραμμή «179» καταχωρείται στη μεταβλητή «diff_terms» η διαφορά του δεύτερου όρου με τον πρώτο, του τρίτου με τον δεύτερο κ.ο.κ. της απειροσειράς, κάθε φορα που εκτελείται ο «1^{ος} βρόχος».

Υπολογισμός της απόλυτης τιμής της διαφοράς των όρων (γραμμή 180)

Στη γραμμή «180» καταχωρείται στη μεταβλητή «abs_diff_terms» η απόλυτη τιμή της διαφοράς του δεύτερου όρου με τον πρώτο, του τρίτου με τον δεύτερο κ.ο.κ της απειροσειράς, κάθε φορά που εκτελείται ο «1^{ος} βρόχος». Η απόλυτη τιμή υπολογίζεται με τη βοήθεια της συνάρτησης «fabs(…)» που εισάγεται με τη βιβλιοθήκη «math.h».

(~) 1η επανάληψη του 1ου βρόχου (γραμμές 181-186)
(~) 2η, 3η ... επανάληψη του 1ου βρόχου (γραμμές 187-191)

Στις γραμμές «181-191» υλοποιείται ένας έλεγχος με την εντολή «if – else» για τον αριθμό επανάληψης του «1^{ου} βρόχου» που βρίσκεται το πρόγραμμα. Η συνθήκη ελέγχου είναι η παράσταση «j == 2» που ελέγχει το περιεχόμενο της βοηθητικής μεταβλητής «j» για τον υπολογισμό του δεύτερου όρου, του τρίτου κ.ο.κ. της απειροσειράς. Η συνθήκη ελέγχου πραγματοποιείται, διότι, οι χαρακτηριστικές πράξεις για τον υπολογισμό του συνημιτόνου μιας γωνίας σε ακτίνια δεν διαφέρουν μεταξύ τους από το δεύτερο άθροισμα και μετά. Προφανώς, το άθροισμα του πρώτου με τον δεύτερο όρο είναι αυτό που διαφέρει ελάχιστα από τα άλλα αθροίσματα.

(~) 1η επανάληψη του 1ου βρόχου (γραμμές 181-186)

Στις γραμμές «181-186» εκτελούνται οι εντολές της «if (j == 2)» (γραμμή 181) για την περίπτωση που η «j» περιέχει την τιμή «2», δηλαδή, ότι το πρόγραμμα βρίσκεται στην πρώτη εκτέλεση του «1ου βρόχου». Συνεπώς, υπολογίζεται το πρώτο άθροισμα της απειροσειράς που διαφέρει ελάχιστα από τα υπόλοιπα. Για «ω» γωνία σε ακτίνια υπολογίζεται η παράσταση «1 - ω^2 / 2!».

Υπολογισμός του πρώτου αθροίσματος με το κατάλληλο πρόσημο "sign" (γραμμή 183)

Στη γραμμή «183» υλοποιείται η παράσταση «term + (sign * next_term)» για το πρώτο άθροισμα (του πρώτου όρου με τον δεύτερο) με το κατάλληλο πρόσημο που καθορίζει η μεταβλητή «sign» που έχει για αρχική τιμή, τη τιμή «-1» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 167)». Το αποτέλεσμα καταχωρείται στη μεταβλητή «first_cos_T».

Καταχώρηση του αθροίσματος στη μεταβλητή "cos_T" (γραμμή 184)

Στη γραμμή «184» καταχωρείται με την παράσταση «cos_T + first_cos_T» το πρώτο άθροισμα με το κατάλληλο πρόσημο στη μεταβλητή «cos_T» που έχει για αρχική τιμή την τιμή «0.0» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 165)» .

Αλλαγή προσήμου (γραμμή 185)

Στη γραμμή «185» καταχωρείται στη μεταβλητή «sign» η χαρακτηριστική παράσταση «sign * (-1)» που βοηθάει στο να αλλάζει το πρόσημο της απειροσειράς αμέσως μετά το πρώτο άθροισμα και μετά.

(~) 2η, 3η ... επανάληψη του 1ου βρόχου (γραμμές 187-191)

Στις γραμμές «187-191» εκτελούνται οι εντολές της «else» (γραμμή 187) που αντιστοιχεί στην «if (j == 2)» (γραμμή 181), για την περίπτωση που η «j» δεν περιέχει την τιμή «2», δηλαδή, ότι το πρόγραμμα βρίσκεται στην δεύτερη εκτέλεση του «1ου βρόχου» και μετά. Συνεπώς υπολογίζεται το δεύτερο άθροισμα, το τρίτο κ.ο.κ. της απειροσειράς. Για «ω» γωνία σε ακτίνια υπολογίζεται η παράσταση «(1 - ω^2 / 2!) + ω^4 / 4! ...».

Υπολογισμός του δεύτερου αθροίσματος, του τρίτου ... με το κατάλληλο πρόσημο "sign" (γραμμή 189)

Στη γραμμή «189» υλοποιείται η παράσταση «(cos_T + (sign * next_term)» για το δεύτερο άθροισμα, το τρίτο κ.ο.κ. της απειροσειράς με το κατάλληλο πρόσημο που καθορίζει η μεταβλητή «sign», όπου η «sign» έχει περιεχόμενο τη τιμή «1» μετά την πρώτη εκτέλεση του «1ου βρόχου». Το αποτέλεσμα

καταχωρείται στη μεταβλητή «cos_T», όπου η «cos_T» έχει περιεχόμενο το πρώτο άθροισμα μετά την πρώτη εκτέλεση του «1ου βρόχου».

Αλλαγή προσήμου (γραμμή 190)

Στη γραμμή «190» καταχωρείται στη μεταβλητή «sign» η χαρακτηριστική παράσταση «sign * (-1)» που βοηθάει στο να αλλάζει το πρόσημο της απειροσειράς αμέσως μετά το δεύτερο άθροισμα και μετά.

Επιστροφή του αθροίσματος των όρων (απειροσειρά Taylor) (γραμμή 195)

Στη γραμμή «195» η εντολή «return cos_T» επιστρέφει το συνολικό άθροισμα των όρων της απειροσειράς, εκεί που είναι ο έλεγχος του προγράμματος.

Print_Cos_TaylorC (ω) (γραμμές 198-206)

Στις γραμμές «198-206» υλοποιείται η συνάρτηση «Print_Cos_TaylorC(…)», τύπου «void» και με παράμετρο τη μεταβλητή «rad_PCTC» (τύπου «double»), όπου δέχεται για είσοδο τη γωνία που εισήχθη, σε ακτίνια, καλεί τις συναρτήσεις «Cos(…)», «Taylor_C(…) και τυπώνει τις τιμές που επιστρέφουν, δηλαδή, το συνημίτονο της γωνίας με την έτοιμη συνάρτηση «cos(...)» και το συνημίτονο με την απειροσειρά «Taylor» αντίστοιχα. Η διάσχιση της «Print_Cos_TaylorC(…) έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 200)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Print_Cos_TaylorC (rad_PCTC)» σελίδες «17-18»

Κλήση της συνάρτησης "Cos (ω)" (γραμμή 202)

Στη γραμμή «202» η «Print_Cos_TaylorC(…)» καλεί τη συνάρτηση «Cos(…)» που επιστρέφει το συνημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση «cos(...)». Το αποτέλεσμα καταχωρείται στη μεταβλητή «rad_Cos».

Εκτύπωση του συνημιτόνου της γωνίας με την έτοιμη συνάρτηση "cos (ω)" (γραμμή 203)

Στη γραμμή «203» τυπώνεται από τη «standard» έξοδο το περιεχόμενο της μεταβλητής «rad_Cos» (το συνημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση «cos()», που επιστρέφει η συνάρτηση «Cos(…)») με μία συνάρτηση «printf()» συνοδευόμενο με το κατάλληλο μήνυμα. Αξίζει να σημειωθεί ότι το αλφαριθμητικό μορφοποίησης είναι το «%20.6lf», δηλαδή, το αποτέλεσμα θα τυπωθεί αμεσώς μετά από «20» κενούς χαρακτήρες με «6» δεκαδικά ψηφία, καθώς, πρόκειται για πραγματικό διπλής ακρίβειας. Αυτό αποσκοπεί στην ομοιόμορφη στοίχιση των αποτελεσμάτων.

Κλήση της συνάρτησης "Taylor_C (ω)" (γραμμή 204)

Στη γραμμή «204» η «Print_Cos_TaylorC(…)» καλεί τη συνάρτηση «Taylor_C(…)» που επιστρέφει το συνημίτονο της γωνίας σε ακτίνια, με τη χαρακτηριστική απειροσειρά «Taylor». Το αποτέλεσμα καταχωρείται στη μεταβλητή «rad_TaylorCos».

Εκτύπωση του συνημιτόνου της γωνίας με την απειροσειρά "Taylor" (γραμμή 205)

Στη γραμμή «205» τυπώνεται από τη «standard» έξοδο το περιεχόμενο της μεταβλητής «rad_TaylorCos» (το συνημίτονο της γωνίας σε ακτίνια με την απειροσειρά «Taylor» που επιστρέφει η συνάρτηση «Taylor_C(…)») με μία συνάρτηση «printf()» συνοδευόμενο με το κατάλληλο μήνυμα. Αξίζει να σημειωθεί ότι το αλφαριθμητικό μορφοποίησης είναι το «%20.6lf», δηλαδή, το αποτέλεσμα θα τυπωθεί αμεσώς μετά από «20» κενούς χαρακτήρες με «6» δεκαδικά ψηφία, καθώς, πρόκειται για πραγματικό διπλής ακρίβειας. Αυτό αποσκοπεί στην ομοιόμορφη στοίχιση των αποτελεσμάτων.

Check_Cos_TaylorC (ω) (γραμμές 208-227)

Στις γραμμές «208-227» υλοποιείται η συνάρτηση «Check_Cos_Taylor_C (...)», τύπου «void» και με παράμετρο τη μεταβλητή «rad_CCTC», όπου δέχεται για είσοδο τη γωνία που εισήχθη, σε ακτίνια, καλεί τις συναρτήσεις «Cos(…)», «Taylor_C(…) και τυπώνει τις τιμές που επιστρέφουν, δηλαδή, το συνημίτονο της γωνίας με την έτοιμη συνάρτηση «cos(...)» και το συνημίτονο με την απειροσειρά «Taylor» αντίστοιχα. Υπολογίζει την απόλυτη τιμή της διαφοράς των δύο αριθμών που υπολογίζουν το συνημίτονο και συγκρίνει αν είναι «σχεδόν» ίσοι ή όχι. Η διάσχιση της «Check_Cos_TaylorC(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμές 210-211)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Check_Cos_TaylorC (rad_CCTC) σελίδα «18»

Κλήση της συνάρτησης "Cos (ω)" (γραμμή 213)

Στη γραμμή «213» η «Check_Cos_TaylorC(…)» καλεί τη συνάρτηση «Cos(…)» που επιστρέφει το συνημίτονο της γωνίας σε ακτίνια με την έτοιμη συνάρτηση «cos(...)». Το αποτέλεσμα καταχωρείται στη μεταβλητή «CheckCos».

Κλήση της συνάρτησης "Taylor_C (ω)" (γραμμή 214)

Στη γραμμή «214» η «Check_Cos_TaylorC(…)» καλεί τη συνάρτηση «Taylor_C(…)» που επιστρέφει το συνημίτονο της γωνίας σε ακτίνια, με τη

χαρακτηριστική απειροσειρά «Taylor». Το αποτέλεσμα καταχωρείται στη μεταβλητή «CheckTaylorC».

Υπολογισμός της διαφοράς της συνάρτησης "cos (ω)" με την απειροσειρά "Taylor" (γραμμή 215)

Στη γραμμή «215» υλοποιείται η παράσταση «CheckCos – CheckTaylorC» για τον υπολογισμό της διαφοράς του συνημιτόνου της γωνίας σε ακτίνια με την έτοιμη συνάρτηση «cos()» και την απειροσειρά «Taylor». Το αποτέλεσμα καταχωρείται στη μεταβλητή «diff_CheckCosTaylorC»

Υπολογισμός της απόλυτης τιμής της διαφοράς της συνάρτησης "cos (ω)" με την απειροσειρά "Taylor" (γραμμή 216)

Στη γραμμή «216» υπολογίζεται η απόλυτη τιμή της διαφοράς των δύο αριθμών που βγάζουν το συνημίτονο της γωνίας. Η απόλυτη τιμή υπολογίζεται με τη βοήθεια της συνάρτησης «fabs(…)» που εισάγεται με τη βιβλιοθήκη «math.h».Το αποτέλεσμα καταχωρείται στη μεταβλητή «abs_diff_CheckCosTaylorC».

(~) Αποδεκτή απόλυτη τιμή της διαφοράς της διαφοράς του συνημιτόνου με την απειροσειρά "Taylor" (γραμμές 217-221)
(~) Απορριπτέα απόλυτη τιμή της διαφοράς της διαφοράς του συνημιτόνου με την απειροσειρά "Taylor" (γραμμές 222-226)

Στις γραμμές «217-226» υλοποιείται ένας έλεγχος με την εντολή «if – else» ως προς την απόκλιση της απόλυτης τιμής της διαφοράς των δύο αριθμών που βγάζουν το συνημίτονο της γωνίας. Η συνθήκη ελέγχου είναι η παράσταση «abs_diff_CheckCosTaylorC <= 0.0000009». Ο έλεγχος πραγματοποιείται για να διαπιστωθεί αν οι δύο αριθμοί είναι «σχεδόν» ίσοι.

(~) Αποδεκτή απόλυτη τιμή της διαφοράς της συνάρτησης "cos (ω)" με την απειροσειρά "Taylor" (γραμμές 217-221)

Στις γραμμές «217-221» εκτελούνται οι εντολές της «if (abs_diff_CheckCosTaylorC)» (γραμμή 121) για την περίπτωση που η απόλυτη τιμή της διαφοράς των δύο αριθμών είναι μικρότερη ή ίση από την τιμή «0.0000009» και επομένως, είναι «σχεδόν» ίσοι. Στις γραμμές «219» και «220» τυπώνονται από τη «standard» έξοδο, με τη συνάρτηση «printf(…)» τα χαρακτηριστικά μηνύματα.

(~) Απορριπτέα απόλυτη τιμή της διαφοράς της συνάρτησης "sin (ω)" με την απειροσειρά "Taylor" (γραμμές 222-226)

Στις γραμμές «222-226» εκτελούνται οι εντολές της «else» που αντιστοιχεί στην «if (abs_diff_CheckCosTaylorC)» (γραμμή 217) για την περίπτωση που η απόλυτη τιμή της διαφοράς των δύο αριθμών είναι μεγαλύτερη από την τιμή

«0.0000009» και επομένως, οι δύο αριθμοί δεν είναι «σχεδόν» ίσοι. Στις γραμμές «224» και «225» τυπώνονται από τη «standard» έξοδο, με τη συνάρτηση «printf(…)» τα χαρακτηριστικά μηνύματα.

ΠΑΡΑΔΕΙΓΜΑΤΑ

Παράδειγμα 1 (Ω == 45°)

==========

Υπολογισμός του ημιτόνου και του συνημιτόνου μίας γωνίας

==========

Εισάγετε γωνία στο διάστημα του 1ου κύκλου [0,360]

Μοίρες : 45

Μοίρες : [45.000000]

Ακτίνια : [0.785397]

Sine : [0.707106]

Taylor : [0.707106]

Sine ~= Taylor

Οι δύο αριθμοί είναι σχεδόν ίσοι

Cosine : [0.707107]

Taylor : [0.707107]

Cosine ~= Taylor

Οι δύο αριθμοί είναι σχεδόν ίσοι

Παράδειγμα 2 (Ω == 180°)

==========

Υπολογισμός του ημιτόνου και του συνημιτόνου μίας γωνίας

==========

Εισάγετε γωνία στο διάστημα του 1ου κύκλου [0,360]

Μοίρες : 180

Μοίρες : [180.000000]

Ακτίνια : [3.141590]

Sine : [0.000003]

Taylor : [0.000003]

Sine ~= Taylor

Οι δύο αριθμοί είναι σχεδόν ίσοι

Cosine : [-1.000000]

Taylor : [-1.000000]

Cosine ~= Taylor

Οι δύο αριθμοί είναι σχεδόν ίσοι

Παράδειγμα 3 (Ω == 167.5)

==========

Υπολογισμός του ημιτόνου και του συνημιτόνου μίας γωνίας

==========

Εισάγετε γωνία στο διάστημα του 1ου κύκλου [0,360]

Μοίρες : 214.8963

Μοίρες : [214.896300]

Ακτίνια : [3.750645]

Sine : [-0.572090]

Taylor : [-0.572090]

Sine ~= Taylor

Οι δύο αριθμοί είναι σχεδόν ίσοι

Cosine : [-0.820191]

Taylor : [-0.820191]

Cosine ~= Taylor

Οι δύο αριθμοί είναι σχεδόν ίσοι

Παρατηρήσεις

Η ορθότητα των αποτελεσμάτων στα παραπάνω παραδείγματα ελέγχθηκε και με την χρήση τυπικής αριθμομηχανής και διαπίστωθηκαν κάποιες παρατηρήσεις και στα τρία παραδείγματα.

Αναλυτικά, έχουμε «3» παραδείγματα που αντιστοιχούν σε γωνία που ανήκει στο διάστημα του πρώτου κύκλου [0,360].

Παράδειγμα 1 (Ω == 45^o)

Στο «Παράδειγμα 1» διαβάζεται από τη «standard» είσοδο η γωνία των «45» μοιρών και ορθά μετατρέπεται σε ακτίνια, υπολογίζεται το ημίτονο της με την έτοιμη συνάρτηση «sin(…)» και την απειροσειρά «Taylor», το συνημίτονο της με την έτοιμη συνάρτηση «cos(…)» και την απειροσερά «Taylor». Οι δύο αριθμοί είναι «σχεδόν» ίσοι ως προς το «7^o» δεκαδικό τους ψηφίο, όπου άμα πάρουμε την απόλυτη τιμή της διαφοράς τους θα διαπιστωθεί ότι είναι μικρότερη από την τιμή «0.0000009». Τα αποτελέσματα :

Μοίρες : 45.000000

Ακτίνια : 0.785397

sin(…) : 0.707106

TaylorS : 0.707106

cos(…) : 0.707107

TaylorC : 0.707107

Η παρατήρηση είναι ότι στο συνημίτονο το αποτέλεσμα που βγάζουν και οι δύο αριθμοί είναι η τιμή 0.707107», ενώ, η τυπική αριθμομηχανή βγάζει ότι το συνημίτονο των «45» μοιρών είναι η τιμή «0.707106».

Παράδειγμα 2 (Ω == 180^o)

Στο «Παράδειγμα 2» διαβάζεται από τη «standard» είσοδο η γωνία των «180» μοιρών και ορθά μετατρέπεται σε ακτίνια, υπολογίζεται το ημίτονο της με την έτοιμη συνάρτηση «sin(…)» και την απειροσειρά «Taylor», το συνημίτονο της με την έτοιμη συνάρτηση «cos(…)» και την απειροσερά «Taylor». Οι δύο αριθμοί είναι «σχεδόν» ίσοι ως προς το «7^o» δεκαδικό τους ψηφίο, όπου άμα πάρουμε την απόλυτη τιμή της διαφοράς τους θα διαπιστωθεί ότι είναι μικρότερη από την τιμή «0.0000009». Τα αποτελέσματα :

Μοίρες : 180.000000

Ακτίνια : 3.141590

sin(…) : 0.000003

TaylorS : 0.000003

cos(…) : -1.000000

TaylorC : -1.000000

Η παρατήρηση είναι ότι στο ημίτονο το αποτέλεσμα που βγάζουν και οι δύο αριθμοί είναι η τιμή «0.000003», ενώ η τυπική αριθμομηχανή βγάζει ότι το ημίτονο των «180» μοιρών είναι η τιμή «0.000000».

Παράδειγμα 3 (Ω == 214.8963°)

Στο «Παράδειγμα 3» διαβάζεται από τη «standard» είσοδο η γωνία των «214.8963» μοιρών και ορθά μετατρέπεται σε ακτίνια, υπολογίζεται το ημίτονο της με την έτοιμη συνάρτηση «sin(…)» και την απειροσειρά «Taylor», το συνημίτονο της με την έτοιμη συνάρτηση «cos(…)» και την απειροσερά «Taylor». Οι δύο αριθμοί είναι «σχεδόν» ίσοι ως προς το «7°» δεκαδικό τους ψηφίο, όπου άμα πάρουμε την απόλυτη τιμή της διαφοράς τους θα διαπιστωθεί ότι είναι μικρότερη από την τιμή «0.0000009». Τα αποτελέσματα :

Μοίρες : 214.896300

Ακτίνια : 3.750645

sin(…) : -0.572090

TaylorS : -0.572090

cos(…) : -0.820191

TaylorC : -0.820191

Η παρατήρηση είναι ότι στο συνημίτονο το αποτέλεσμα που βγάζουν και οι δύο αριθμοί είναι η τιμή «-0.820191», ενώ η τυπική αριθμομηχανή βγάζει ότι το ημίτονο των «180» μοιρών είναι η τιμή «-0.820188».

ΘΕΜΑ 2

ΕΠΙΣΗΜΑΝΣΗ «Menu.c»

Το «Πρόγραμμα “Menu.c”» (Πηγαίος Κώδικας) και η «Τεκμηρίωση “Menu.c” (Ζητούμενο, Δομή, Συναρτήσεις, Μεταβλητές, Διάσχιση, Παραδείγματα) απαντούν στο ζητούμενο του ερωτήματος «Θέμα 2».

ΠΡΟΓΡΑΜΜΑ «Menu.c»

```
1   #include <stdio.h>
2   #include <math.h>
3   /* Δήλωση συναρτήσεων */
4   void Title (); // Ο τίτλος του προγράμματος
5   int Read_A (); // Εισαγωγή του πρώτου ακεραίου αριθμού "A"
6   int Read_B (); // Εισαγωγή του δεύτερου ακεραίου αριθμού "B"
7   float Power (int, int); // Υπολογισμός της δύναμης "A^B" (λειτουργία [1])
8    void Check_Power (int, int); // Έλεγχος εγκυρότητας της λειτουργίας [1] και εκτύπωση των αντίστοιχων αποτελεσμάτων
9    int Check_Valid_Power (int, int); // Καταμέτρηση της έγκυρης λειτουργίας [1]
10  int Factorial (int); // Υπολογισμός των παραγοντικών "A!" και "B!" (λειτουργία [2])
11  void Check_Factorial_A (int); // Έλεγχος εγκυρότητας της υπολειτουργίας "A!"
12  void Check_Factorial_B (int); // Έλεγχος εγκυρότητας της υπολειτουργίας "B!"
13  void Check_Factorial (int, int); // Έλεγχος εγκυρότητας της λειτουργίας [2] και εκτύπωση των αντίστοιχων αποτελεσμάτων
14  int Check_Valid_Factorial (int, int); // Καταμέτρηση της έγκυρης λειτουργίας [2]
15  int Combinations (int, int); // Υπολογισμός του πλήθους των συνδυασμών "A" ανά "B" (λειτουργία [3])
16  void Check_Combinations (int, int); // Έλεγχος εγκυρότητας της λειτουργίας [3] και εκτύπωση τον αντίστοιχων αποτελεσμάτων
17   int Check_Valid_Combinations (int, int); // Καταμέτρηση της έγκυρης λειτουργίας [3]
18  int Exit (); // Καταμέτρηση της έγκυρης λειτουργίας [4] (Έξοδος)
19  void Menu (int, int); // Το μενού για την επιλογή των λειτουργίων
20 /* Όπου "A" ο πρώτος ακέραιος αριθμός και "B" ο δεύτερος ακέραιος αριθμός */
21
22 int main (int argc, char **argv) /* main (argc, **argv) */
```

```
23 {
24      system ("chcp 1253");
25
26      int a, b; // Δήλωση μεταβλητών
27
28      Title (); // Κλήση της συνάρτησης "Title ()"
29      a = Input_A (); // Κλήση της συνάρτησης "Input_A ()"
30      b = Input_B (); // Κλήση της συνάρτησης "Input_B ()"
31      Menu (a, b); // Κλήση της συνάρτησης "Menu (A, B)"
32
33      return 0;
34 }
35
36 void Title () /* Title () */
37 {
38     printf ("====================================\n\n");
39     printf ("Μενού αριθμητικών πράξεων\n\n"); // Τίτλος προγράμματος
40     printf ("====================================\n\n");
41 }
42
43 int Read_A () /* Read_A (A) */
44 {
45      int a_RA; // Δήλωση μεταβλητών
46
47      printf ("Εισάγετε τον ακέραιο αριθμό A : ");
48      scanf ("%d", &a_RA); // Εισαγωγή του ακεραίου αριθμού "A"
49
50      return a_RA; // Επιστροφή του ακεραίου αριθμού "A"
51 }
52
53 int Read_B () /* Read_B (B) */
54 {
55      int b_RB; // Δήλωση μεταβλητών
56
57      printf ("Εισάγετε τον ακέραιο αριθμό B : ");
58      scanf ("%d", &b_RB); // Εισαγωγή του ακεραίου αριθμού "B"
```

```
59
60      return b_RB; // Επιστροφή του ακεραίου αριθμού "B"
61 }
62
63 float Power (int a_P, int b_P) /* Power (A, B) */
64 {
65      float power; // Δήλωση μεταβλητών
66      float error_P = -1.0; // Αρχικοποίηση μεταβλητών
67
68      if (a_P == 0.0 && b_P == 0.0) /* (~) A == 0 KAI B == 0 */
69      {
70            return error_P; // Επιστροφή τιμής σφάλματος
71      }
72      else /* (~) A != 0 KAI B != 0 */
73      {
74             power = pow (a_P,  b_P); // Υπολογισμός της τιμής της δύναμης
"A^B"
75
76             return power; // Επιστροφή της τιμής της δύναμης "A^B"
77      }
78 }
79
80 void Check_Power (int a_CP, int b_CP) /* Check_Power (A, B) */
81 {
82      system ("cls");
83
84      float cp; // Δήλωση μεταβλητών
85
86      printf ("[1] Υπολογισμός της δύναμης A^B\n");
87      cp = Power (a_CP, b_CP); // Κλήση της συνάρτησης "Power (A, B)"
88      if (cp != -1) /* (~) Μη τιμή σφάλματος */
89      {
90             printf ("A^B : %f\n\n", cp); // Εκτύπωση του αποτελέσματος της
λειτουργίας [1] (A^B)
91             printf ("-----------------------------------------\n\n");
92       }
93      else /* (~) Τιμή σφάλματος */
94      {
```

```
95              printf ("Σφάλμα\n");
96              printf ("-----------------------------------------\n\n");
97      }
98 }
99
100 int Check_Valid_Power (int a_CVP, int b_CVP) /* Check_Valid_Power (A, B)
*/
101 {
102     float cvp; // Δήλωση μεταβλητών
103
104     cvp = Power (a_CVP, b_CVP); // Κλήση της συνάρτησης "Power (A, B)"
105     if (cvp != -1) /* (~) Μη τιμή σφάλματος */
106     {
107             return 1; // Επιστροφή τιμής έγκυρης λειτουργίας
108     }
109     else /* (~) Τιμή σφάλματος */
110     {
111             return 0; // Επιστροφή τιμής άκυρης λειτουργίας
112     }
113 }
114
115 int Factorial (int a_b_F) /* Factorial (A_B) */
116 {
117     int i; // Δήλωση μεταβλητών
118     int p = 1; // Αρχικοποίηση μεταβλητών
119     int error_F = -1;
120
121     if (x_y_F >= 0) /* (~) A_B >= 0 */
122     {
123             for (i = 1 ; i <= x_y_F ; i++) /* Βρόχος */
124             {
125                     p = p * i; // Υπολογισμός της τιμής του παραγοντικού "A!"
ή του "B!"
```

```
126             }
127
128             return p; // Επιστροφή της τιμής του παραγοντικού "A!" ή του
"B!"
129     }
130     else /* (~) A_B < 0 */
131     {
132             return error_F; // Επιστροφή τιμής σφάλματος
133     }
134 }
135
136 void Check_Factorial_A (int a_CFA) /* Check_Factorial_A (A) */
137 {
138     int cf_a;
139
140     cf_a = Factorial (a_CFA); // Κλήση της συνάρτησης "Factorial (A)"
141     if (cf_a != -1) /* (~) Μη τιμής σφάλματος */
142     {
143             printf ("A! : [%20d]\n", cfa); // Εκτύπωση του αποτελέσματος της
υπολειτουργίας [2] (A!)
144     }
145     else /* (~) Τιμή σφάλματος */
146     {
147             printf ("Σφάλμα\n");
148     }
149 }
150
151 void Check_Factorial_B (int b_CFB) /* Check_Factorial_B (B) */
152 {
153     int cf_b; // Δήλωση μεταβλητών
154
155     cf_b = Factorial (b_CFB); // Κλήση της συνάρτησης "Factorial (B)"
156     if (cf_b != -1) /* (~) Μη τιμής σφάλματος */
157     {
158             printf ("B! : [%20d]\n\n", cf_b); // Εκτύπωση του αποτελέσματος
της υπολειτουργίας [2] (B!)
159             printf  ("-----------------------------------------\n\n");
160     }
```

```
161     else  /* (~) Τιμή σφάλματος */
162     {
163             printf ("Σφάλμα\n\n");
164             printf  ("-----------------------------------------\n\n");
165     }
166 }
167
168 void Check_Factorial (int a_CF, int b_CF) /* Check_Factorial (A, B) */
169 {
170     system ("cls");
171
172     printf ("[2] Υπολογισμός του A! και του B!\n");
173     Check_Factorial_A (a_CF); // Κλήση της συνάρτησης "Check_Factorial_A
(A)"
174     Check_Factorial_B (b_CF); // Κλήση της συνάρτησης "Check_Factorial_B
(B)"
175 }
176
177 int Check_Valid_Factorial (int a_CVF, int b_CVF) /* Check_Valid_Factorial
(A, B) */
178 {
179     int cvf_a, cvf_b; // Δήλωση μεταβλητών
180
181     cvf_a = Factorial (a_CVF); // Κλήση της συνάρτησης "Factorial (A)"
182     cvf_b = Factorial (b_CVF); // Κλήση της συνάρτησης "Factorial (A)"
183     if (cvf_a != -1 && cvf_b != -1) /* (~) Μη τιμή σφάλματος */
184     {
185             return 1; // Επιστροφή τιμής έγκυρης λειτουργίας
186     }
187     else  /* (~) Τιμή σφάλματος */
188     {
189             return 0; // Επιστροφή τιμής άκυρης λειτουργίας
190     }
191 }
192
193 int Combinations (int a_C, int b_C) /* Combinations (A, B) */
194 {
```

```
195     int combos, i, j, k; // Δήλωση μεταβλητών
196     int error_C = -1; // Αρχικοποίηση μεταβλητών μεταβλητών
197
198     if (a_C > b_C && a_C >= 0 && b_C >= 0) /* (~) A > B ΚΑΙ A >= 0 ΚΑΙ B
>= 0*/
199     {
200             i = Factorial (a_C); // Κλήση της συνάρτησης "Factorial (A)"
201             j = Factorial (b_C); // Κλήση της συνάρτησης "Factorial (B)"
202             k = Factorial (a_C - b_C); // Κλήση της συνάρτησης "Factorial
(A - B)"
203             combos =  i / (j * k); // Υπολογισμός του πλήθους των συνδυασμών
"A" ανά "B"
204
205             return combos; // Επιστροφή του πλήθους των συνδυασμών "A" ανά
"B"
206     }
207     else /* (~) A <= B Ή A < 0 Ή B < 0 */
208     {
209             return error_C; // Επιστροφή τιμής σφάλματος
210     }
211 }
212
213 void Check_Combinations (int a_CC, int b_CC) /* Check_Combinations (A, B)
*/
214 {
215    system ("cls");
216
217    int cc; // Δήλωση μεταβλητών
218
219     printf ("[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B\n");
220     cc = Combinations (a_CC, b_CC); // Κλήση της συνάρτησης "Combinations
(A, B)"
221     if (cc != -1) /* (~) Μη τιμή σφάλματος */
222     {
223             printf ("A άνα B : %d\n\n", cc); // Εκτύπωση του αποτελέσματος
της λειτουργίας [3] (A! / B! * (A - B)!)
224             printf  ("-----------------------------------------------\n\n");

225     }
226     else /* (~) Τιμή σφάλματος */
227     {
```

```
228             printf ("Σφάλμα\n\n");
229             printf  ("-----------------------------------------\n\n");
230     }
231 }
232
233 int Check_Valid_Combinations (int a_CVC, int b_CVC)
/* Check_Valid_Combinations (A, B) */
234 {
235     int cvc; // Δήλωση μεταβλητών
236
237     cvc   =   Combinations   (a_CVC,   b_CVC);   //   Κλήση   της   συνάρτησης
"Combinations (A, B)"
238     if (cvc != -1) /* (~) Μη τιμή σφάλματος */
239     {
240             return 1; // Επιστροφή τιμής έγκυρης λειτουργίας
241     }
242     else  /* (~) Τιμή σφάλματος */
243     {
244             return 0; // Επιστροφή τιμής άκυρης λειτουργίας
245    }
246 }
247
248 int Exit () /* Exit () */
249 {
250     int cnt = 1; // Δήλωση και αρχικοποίηση μεταβλητών
251
252     return cnt; // Επιστροφή τιμής έγκυρης λειτουργίας [4]
253 }
254
255 void Menu (int a_M, int b_M) /* Menu (A, B) */
256 {
257    system ("cls");
258
259    int ch, sum; // Δήλωση μεταβλητών
260    int m_P = 0; // Αρχικοποίηση μεταβλητών
261    int m_F = 0; // Αρχικοποίηση μεταβλητών
262    int m_C = 0; // Αρχικοποίηση μεταβλητών
```

```
263    int m_E = 0; // Αρχικοποίηση μεταβλητών
264
265    do /* Βρόχος */
266     {
267            printf ("[1] Υπολογισμός της δύναμης A^B\n"); // Το μενού με τις επιλογές λειτουργιών
268            printf ("[2] Υπολογισμός του A! και του B!\n");
269            printf ("[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B\n");
270            printf ("[4] Έξοδος\n");
271            printf ("\nΕπιλογή λειτουργίας : ");
272            scanf ("%d", &ch); // Εισαγωγή επιλογής λειτουργίας
273            if (ch >= 1 && ch <= 4) /* (~) Έγκυρη επιλογή λειτουργίας */
274            {
275                  switch (ch) /* Οι επιλογές λειτουργιών */
276                  {
277                        case 1 : Check_Power (a_M, b_M); // [1] Κλήση της συνάρτησης "Check_Power (A, B)"
278                        m_P = m_P + Check_Valid_Power (a_M, b_M); break; // Καταμέτρηση της τιμής έγκυρης λειτουργίας [1]
279                        case 2 : Check_Factorial (a_M, b_M); // [2] Κλήση της συνάρτησης "Check_Factorial (A, B)"
280                        m_F = m_F + Check_Valid_Factorial (a_M, b_M); break; // Καταμέτρηση της τιμής έγκυρης λειτουργίας [2]
281                        case 3 : Check_Combinations (a_M, b_M); // [3] Κλήση της συνάρτησης "Combinations (A, B)"
282                        m_C = m_C + Check_Valid_Combinations (a_M, b_M); break; // Καταμέτρηση της τιμής έγκυρης λειτουργίας [3]
283                  }
284              }
285              else /* (~) Άκυρη επιλογή λειτουργίας */
286              {
287                    system ("cls");
288              }
289     }
290     while (ch != 4); // Συνθήκη τερματισμού του βρόχου
291
292     system ("cls");
293
294     m_E = Exit (); // [4] Κλήση της συνάρτησης "Exit ()"
```

```
295     sum = m_P + m_F + m_C + m_E; // Υπολογισμός του πλήθους των έγκυρων λειτουργιών

296     printf ("\nΑριθμός έγκυρων λειτουργιών : %d\n", sum); // Εκτύπωση του του πλήθους των έγκυρων λειτουργιών

297 }
```

ΤΕΚΜΗΡΙΩΣΗ «Menu.c»

ΖΗΤΟΥΜΕΝΟ

Το πρόγραμμα «Menu.c» επιτυγχάνει τις εξής λειτουργίες:

a) Διαβάζει από τη «standard» είσοδο δύο ακέραιους αριθμούς «A» και «B».

b) Διαβάζει από ένα μενού, έναν ακέραιο αριθμό που αντιστοιχεί σε μία λειτουργία που υλοποιεί διάφορες μαθηματικές πράξεις με τους δύο ακέραιους αριθμούς «A», «B», εφόσον, δεν υπάρχει κάποιος περιορισμός.

c) Υπολογίζει τη δύναμη «A^B», εφόσον, δεν υπάρχει κάποιος περιορισμός.

d) Τυπώνει στη «standard» έξοδο το αποτέλεσμα της δύναμης «A^B», εφόσον, δεν υπάρχει κάποιος περιορισμός.

e) Υπολογίζει το παραγοντικό «A!» και το παραγοντικό «B!», εφόσον, δεν υπάρχει κάποιος περιορισμός.

f) Τυπώνει στη «standard» έξοδο το αποτέλεσμα του παραγοντικού «A!» και του παραγοντικού «B!», εφόσον, δεν υπάρχει κάποιος περιορισμός.

g) Υπολογίζει το πλήθος των συνδυασμών «A» ανά «B», εφόσον, δεν υπάρχει κάποιος περιορισμός.

h) Τυπώνει στη «standard» έξοδο το αποτέλεσμα του πλήθους των συνδυασμών «A» ανά «B», εφόσον, δεν υπάρχει κάποιος περιορισμός.

i) Υπολογίζει το πλήθος των έγκυρων λειτουργιών και το τυπώνει στη «standard» έξοδο, συμπεριλαμβάνοντας και την λειτουργία της εξόδου.

j) Σε περίπτωση που η λειτουργία είναι άκυρη τυπώνεται το στη «standard» έξοδο το χαρακτηριστικό μήνυμα.

ΔΟΜΗ

Προκειμένου να υλοποιηθεί το ζητούμενο χρησιμοποίηθηκαν, αρχικά, οι βιβλιοθήκες (.h) :

a) «stdio.h»: Περιέχει τις έτοιμες συναρτήσεις «scanf(...)» και «printf(...)» που συνδέονται με τα κανάλια εισόδου και εξόδου αντίστοιχα για την ανάγνωση και την τύπωση περιεχομένων των αντίστοιχων μεταβλητών. Επίσης, η «printf(...)» χρησιμοποιήθηκε για να τυπωθούν χαρακτηριστικά μηνύματα για την βέλτιστη κατανόηση του πηγαίου κώδικα.

b) «math.h»: Περιέχει την έτοιμη συνάρτηση «pow(…)» για τον υπολογισμό της δύναμης «A^B».

Επιπρόσθετα, χρησιμοποιήθηκαν οι χαρακτηριστικοί τελεστές :

a) αριθμητικοί : +, -, *, /

b) σχεσιακοί : <=, >=, !=,

c) λογικοί : &&

d) ανάθεσης : =

e) τελεστής & : Για την διεύθυνση μεταβλητής ως δεύτερο όρισμα της συνάρτησης «scanf()» που συνδέεται με τη «standard» είσοδο

f) μετααύξησης : μεταβλητή++

Οι εντολές ελέγχου :

a) if – else

Οι εντολές επαναλήψης :

a) do – while

b) for

Η κάθε λειτουργία απ’ την ενότητα «Ζητούμενο» υλοποιήθηκε με αυτόνομα υποπρογράμματα (βλ. ενότητα «Συναρτήσεις»).

ΣΥΝΑΡΤΗΣΕΙΣ

Τύπου «void» (δεν επιστρέφουν τιμή)

Title () «Ο τίτλος του προγράμματος»

Check_Power (a_CP, b_CP) «Έλεγχος εγκυρότητας της λειτουργίας [1] και εκτύπωση των αντίστοιχων αποτελεσμάτων»

Check_Factorial_A (a_CF) «Έλεγχος εγκυρότητας της υπολειτουργίας "A!"»

Check_Factorial_B (b_CF) «Έλεγχος εγκυρότητας της υπολειτουργίας "B!"»

Check_Combinations (a_CC, b_CC) «Έλεγχος εγκυρότητας της λειτουργίας [3] και εκτύπωση τον αντίστοιχων αποτελεσμάτων»

Check_Factorial (a_CF, b_CF) «Έλεγχος εγκυρότητας της λειτουργίας [2] και εκτύπωση των αντίστοιχων αποτελεσμάτων»

Menu (a_M, b_M) «Το μενού για την επιλογή των λειτουργίων»

Τύπου «int» (επιστρέφουν ακέραια τιμή)

Read_A () «Εισαγωγή του πρώτου ακεραίου αριθμού "A"»

Read_B () «Εισαγωγή του δευτέρου ακεραίου αριθμού "B"»

Check_Valid_Power (a_CVP, b_CVP) «Καταμέτρηση της έγκυρης λειτουργίας [1]»

Factorial (a_b_F) «Υπολογισμός των παραγοντικών "A!" και "B!" (λειτουργία [2])»

Check_Valid_Factorial (a_CVF, b_CVF) «Καταμέτρηση της έγκυρης λειτουργίας [2]»

Combinations (a_C, b_C) «Υπολογισμός του πλήθους των συνδυασμών "A" ανά "B" (λειτουργία [3])»

Check_Valid_Combinations (a_CVC, b_CVC) «Καταμέτρηση της έγκυρης λειτουργίας [3]»

Exit () «Καταμέτρηση της έγκυρης λειτουργίας [4] (Έξοδος)»

Τύπου «float» (επιστρέφουν πραγματική τιμή)

5Power (a_P, b_P) «Υπολογισμός της δύναμης "A^B" (λειτουργία [1])»

ΜΕΤΑΒΛΗΤΕΣ

main (int argc, char **argv)

Ακέραιες μεταβλητές (τύπου «int»)

a (Ο πρώτος ακέραιος αριθμός «Α»)

b (Ο δεύτερος ακέραιος αριθμός «Β»)

Read_A ()

Ακέραιες μεταβλητές (τύπου «int»)

a_RA (Ο πρώτος ακέραιος αριθμός «Α»)

Read_B ()

Ακέραιες μεταβλητές (τύπου «int»)

b_RB (Ο δεύτερος ακέραιος αριθμός «Β»)

Power (a_P, b_P)

Παράμετροι

a_P (τύπου «int» / Ο πρώτος ακέραιος αριθμός «Α»)

b_P (τύπου «int» / Ο δεύτερος ακέραιος αριθμός «Β»)

Πραγματικές μεταβλητές (τύπου «float»)

power (Η τιμή της δύναμης «A^B»)

error_P (Τιμή σφάλματος)

Check_Power (a_CP, b_CP)

Παράμετροι

a_CP (τύπου «int» / Ο πρώτος ακέραιος αριθμός «Α»)

b_CP (τύπου «int» / Ο δεύτερος ακέραιος αριθμός «Β»)

Πραγματικές μεταβλητές (τύπου «float»)

cp (Η τιμή που επέστρεψε η «Power(…)»)

Check_Valid_Power (a_CVP, b_CVP)

Παράμετροι

a_CVP (τύπου «int» / Ο πρώτος ακέραιος αριθμός «A»)

b_CVP (τύπου «int» / Ο δεύτερος ακέραιος αριθμός «B»)

Πραγματικές μεταβλητές (τύπου «float»)

cvp (Η τιμή εγκυρότητας που επέστρεψε η «Power(…)»)

Factorial (a_b_F)

Παράμετροι

a_b_F (τύπου «int» / Ο πρώτος ακέραιος αριθμός «A» ή ο δεύτερος ακέραιος αριθμός «B»)

Ακέραιες μεταβλητές (τύπου «int»)

i (Ο μετρητής που ελέγχει το βρόχο)

p (Το παραγοντικό «A!» ή «B!»)

error_F (Τιμή σφάλματος)

Check_Factorial_A (a_CFA)

Παράμετροι

a_CFA (τύπου «int» / Ο πρώτος ακέραιος αριθμός «A»)

Ακέραιες μεταβλητές (τύπου «int»)

cf_a (Η τιμή που επιστρέφει η «Factorial(…)» για το «A»)

Check_Factorial_B (b_CFB)

Παράμετροι

b_CFB (τύπου «int» / Ο πρώτος ακέραιος αριθμός «Β»)

Ακέραιες μεταβλητές (τύπου «int»)

cf_b (Η τιμή που επιστρέφει η «Factorial(…)» για το «Β!»)

Check_Factorial (a_CF, b_CF)

Παράμετροι

a_CF (τύπου «int» / Ο πρώτος ακέραιος αριθμός «Α»)

b_CF (τύπου «int» / Ο δεύτερος ακέραιος αριθμός «Β»)

Check_Valid_Factorial (a_CVF, b_CVF)

Παράμετροι

a_CVF (τύπου «int» / Ο πρώτος ακέραιος αριθμός «Α»)

b_CVF (τύπου «int» / Ο δεύτερος ακέραιος αριθμός «Β»)

Ακέραιες μεταβλητές (τύπου «int»)

cvf_a (Η τιμή που επιστρέφει η «Factorial(…)» για το «Α!»)

cvf_b (Η τιμή που επιστρέφει η «Factorial(…)» για το «Β!»)

Combinations (a_C, b_C)

Παράμετροι

a_C (τύπου «int» / Ο πρώτος ακέραιος αριθμός «Α»)

b_C (τύπου «int» / Ο δεύτερος ακέραιος αριθμός «Β»)

Ακέραιες μεταβλητές (τύπου «int»)

i (Η τιμή που επιστρέφει η «Factorial(…)» για το «Α!»)

j (Η τιμή που επιστρέφει η «Factorial(…)» για το «Β!»)

k (Η τίμη που επιστρέφει η «Factorial(…)» για το «(Α – Β)!»)

combos (Το πλήθος των συνδυασμών «Α» ανά «Β»)

error_C (Τιμή σφάλματος)

Check_Combinations (a_CC, b_CC)

Παράμετροι

x_CC (τύπου «int» / Ο πρώτος ακέραιος αριθμός «Α»)

y_CC (τύπου «int» / Ο δεύτερος ακέραιος αριθμός «Β»)

Ακέραιες μεταβλητές (τύπου «int»)

cc (Η τιμή που επιστρέφει η «Combinations(…)»)

Check_Valid_Combinations (a_CVC, b_CVC)

Παράμετροι

a_CVC (τύπου «int» / Ο πρώτος ακέραιος αριθμός «Α»)

b_CVC (τύπου «int» / Ο δεύτερος ακέραιος αριθμός «Β»)

Ακέραιες μεταβλητές (τύπου «int»)

cvc (Η τιμή που επιστρέφει η «Combinations(…)»)

Exit ()

Ακέραιες μεταβλητές (τύπου «int»)

cnt (Τιμή έγκυρης λειτουργίας της εξόδου)

Menu (a_M, b_M)

Παράμετροι

a_M (τύπου «int» / Ο πρώτος ακέραιος αριθμός «Α»)

b_M (τύπου «int» / Ο δεύτερος ακέραιος αριθμός «Β»)

Ακέραιες μεταβλητές (τύπου «int»)

ch (Η έγκυρη τιμή επιλογή λειτουργίας)

sum (Το άθροισμα του πλήθους των έγκυρων λειτουργιών)

m_P (Το πλήθος των έγκυρων λειτουργιών της λειτουργίας [1] της δύναμης «A^B»)

m_F (Το πλήθος των έγκυρων λειτουργιών της λειτουργίας [2] των παραγοντικών «A!» και «B!»)

m_C (Το πλήθος των έγκυρων λειτουργιών της λειτουργίας [3] του πλήθους των συνδυασμών «A!» άνα «B!»)

m_E (Το πλήθος των έγκυρων λειτουργιών της λειτουργίας [4] της εξόδου)

ΔΙΑΣΧΙΣΗ

Δήλωση συναρτήσεων (γραμμές 4-19)

Βλ. ενότητα «Συναρτήσεις» σελίδες «52-53»

main (int argc, char **argv) (γραμμές 22-34)

Στις γραμμές «22-34» υλοποιείται η κύρια συνάρτηση «main(…)», τύπου «int», του προγράμματος που επιστρέφει τον ακέραιο αριθμό «0», όταν το πρόγραμμα τρέχει χωρίς κανένα πρόβλημα. Η διάσχιση της «main(...)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 26)

Βλ. ενότητα «Μεταβλητές», υποενότητα «main (int argc, char **argv) σελίδα «53».

Κλήση της συνάρτησης "Title ()" (γραμμή 28)

Στη γραμμή «28», η «main(...)» καλεί τη συνάρτηση «Title(...)» που υλοποιείται στις γραμμές «36-41».

Κλήση της συνάρτησης "Read_A ()" (γραμμή 29)

Στη γραμμή «29», η «main(...)» καλεί τη συνάρτηση «Read_A(...)» που υλοποιείται στις γραμμές «43-51» και η ακέραια τιμή που επιστρέφει η δεύτερη, καταχωρείται στην ακέραια μεταβλητή «a».

Κλήση της συνάρτησης "Read_B ()" (γραμμή 30)

Στη γραμμή «30», η «main(...)» καλεί τη συνάρτηση «Read_B(...)» που υλοποιείται στις γραμμές «53-61» και η ακέραια τιμή που επιστρέφει η δεύτερη, καταχωρείται στην ακέραια μεταβλητή «b».

Κλήση της συνάρτησης "Menu (A, B)" (γραμμή 31)

Στη γραμμή «31» η «main(…)» καλεί τη συνάρτηση «Menu(…)» που υλοποιείται στις γραμμές «255-297».

Title () (γραμμές 36-41)

Στις γραμμές «36-41» υλοποιείται η συνάρτηση «Title(…), τύπου «void», που τυπώνει τον τίτλο του προγράμματος. Η διάσχιση της «Title(…)» έχει ώς εξής :

Τίτλος προγράμματος (γραμμή 39)

Στη γραμμή «39» τυπώνεται στη «standard» έξοδο με μία συνάρτηση «printf(…)» το μήνυμα «Μενού αριθμητικών πράξεων με ακέραιους» με δύο χαρακτήρες διαφυγής της αλλαγής γραμμής (\n), που απεικονίζουν τον τίτλο του προγράμματος.

Read_A (A) (γραμμές 43-51)

Στις γραμμές «43-51» υλοποιείται η συνάρτηση «Read_A(…)», τύπου «int», όπου διαβάζει από τη «standard» είσοδο τον πρώτο ακέραιο αριθμό «A» και τον επιστρέφει εκεί που είναι ο έλεγχος του προγράμματος. Η διάσχιση της «Read_A(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 45)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Read_A ()» σελίδα «53».

Εισαγωγή του ακεραίου αριθμού "A" (γραμμή 48)

Στη γραμμή «48» διαβάζεται από τη «standard» είσοδο με μία συνάρτηση «scanf(…)» ο πρώτος ακέραιος αριθμός «A» και καταχωρείται στη μεταβλητή «a_RA». Στη γραμμή «47» τυπώνεται στη «standard» έξοδο με μία συνάρτηση «printf(…)» το χαρακτηριστικό μήνυμα.

Επιστροφή του ακεραίου αριθμού "A" (γραμμή 50)

Στη γραμμή «50» η εντολή «return a_RA» επιστρέφει το περιεχόμενο της μεταβλητής «a_RA», δηλαδή, τον πρώτο ακέραιο αριθμό «A», εκεί που είναι ο έλεγχος του προγράμματος.

Read_B (B) **(γραμμές 53-61)**

Στις γραμμές «53-61» υλοποιείται η συνάρτηση «Read_B(…)», τύπου «int», όπου διαβάζει από τη «standard» είσοδο τον δεύτερο ακέραιο αριθμό «B» και τον επιστρέφει εκεί που είναι ο έλεγχος του προγράμματος. Η διάσχιση της «Read_B(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 55)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Read_B ()» σελίδα «53»

Εισαγωγή του ακεραίου αριθμού "B" (γραμμή 58)

Στη γραμμή «58» διαβάζεται από τη «standard» είσοδο με μία συνάρτηση «scanf(…)» ο δεύτερος ακέραιος αριθμός «B» και καταχωρείται στη μεταβλητή «b_RB». Στη γραμμή «57» τυπώνεται στη «standard» έξοδο με μία συνάρτηση «printf(…)» το χαρακτηριστικό μήνυμα.

Επιστροφή του ακεραίου αριθμού "B" (γραμμή 60)

Στη γραμμή «60» η εντολή «return b_RB» επιστρέφει το περιεχόμενο της μεταβλητής «b_RB», δηλαδή, τον δεύτερο ακέραιο αριθμό «B», εκεί που είναι ο έλεγχος του προγράμματος.

Power (A, B) **(γραμμές 63-78)**

Στις γραμμές «63-78» υλοποιείται η συνάρτηση «Power(…)», τύπου «float» και με παραμέτρους τις μεταβλητές «a_P», «b_P» (τύπου «int»), όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» και «B», υπολογίζει τη δύναμη «A^B» και επιστρέφει το αποτέλεσμα ή μία τιμή σφάλματος, εφόσον, υπάρχει κάποιος περιορισμός. Η διάσχιση της «Power(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμές 65-66)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Power (int a_P, int b_P)» σελίδα «53»

Αρχικοποίηση μεταβλητών (γραμμή 66)

Στη γραμμή «66» γίνεται μία αρχικοποίηση της μεταβλητής «error_P» (τιμή σφάλματος), όπου καταχωρείται η τιμή «-1.0».

(~) A == 0 ΚΑΙ B == 0 (γραμμές 68-71)
(~) A != 0 ΚΑΙ B != 0 (γραμμές 72-77)

Στις γραμμές «66-77» υλοποιείται ένας έλεγχος με την εντολή «if – else», όσον αφορά τον περιορισμό που δεν καθιστά εφικτό τον υπολογισμό της

δύναμης «A^B». Η συνθήκη είναι η «a_P == 0.0 && b_P == 0.0» και άμα παράγει μία τιμή «True», τότε θα εκτελεστούν οι εντολές της υποενότητας «(~) A == 0 ΚΑΙ B == 0 (γραμμές 68-71)». Αντιθέτως, θα εκτελεστούν οι εντολές της υποενότητας «(~) A != 0 ΚΑΙ B = 0 (γραμμές 72-77)». Η διάσχιση του ελέγχου έχει ως εξής :

(~) A == 0 ΚΑΙ B == 0 (γραμμές 68-71)

Στις γραμμές «68-71» εκτελούνται οι εντολές της «if (a_P == 0.0 && b_P == 0.0)» (γραμμή 68), για την περίπτωση όπου ο «A» και ο «B» περιέχουν και οι δύο την τιμή «0». Δεν προσδιορίζεται η δύναμη «0^0», συνεπώς, η συνάρτηση επιστρέφει μία τιμή «σφάλματος» που αναλύεται αμέσως παρακάτω. Η διάσχιση της «if» έχει ως εξής :

Επιστροφή τιμής σφάλματος (γραμμή 70)

Στη γραμμή «70» η εντολή «return error_P» επιστρέφει το περιεχόμενο της «error_P» (τιμή σφάλματος), δηλαδή, την τιμή «-1» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 66), εκεί που είναι ο έλεγχος του προγράμματος.

(~) A != 0 ΚΑΙ B != 0 (γραμμές 72-77)

Στις γραμμές «72-77» εκτελούνται οι εντολές της «else» (γραμμή 72) που αντιστοιχεί στην «if (a_P == 0.0 && b_P == 0.0)» (γραμμή 68), για την περίπτωση που ο «A» και ο «B» δεν περιέχουν και οι δύο την τιμή «0». Η διάσχιση της «else» έχει ως εξής :

Υπολογισμός της τιμής της δύναμης "A^B" (γραμμή 74)

Στη γραμμή «74» υπολογίζεται η δύναμη «A^B» που εκτελείται με τη συνάρτηση «pow(…)» που εισάγεται με τη βιβλιοθήκη «math.h». Το αποτέλεσμα καταχωρείται στη μεταβλητή «power».

Επιστροφή της τιμής της δύναμης "A^B" (γραμμή 76)

Στη γραμμή «76» η εντολή «return power» επιστρέφει το περιεχόμενο της «power», δηλαδή, το αποτέλεσμα της δύναμης «A^B», εκεί που είναι ο έλεγχος του προγράμματος.

Check_Power (A, B) (γραμμές 80-98)

Στις γραμμές «80-98» υλοποιείται η συνάρτηση «Check_Power(…)», τύπου «void» και με παραμέτρους τις μεταβλητές «a_CP», «b_CP» (τύπου «int»), όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» και «B», καλεί τη συνάρτηση «Power(…)» και ανάλογα με την τιμή που επέστρεψε τυπώνει στη «standard» έξοδο τα χαρακτηριστικά μηνύματα. Στη γραμμή «86» τυπώνεται

στη «standard» έξοδο το χαρακτηριστικό μήνυμα για την λειτουργία «1». Η διάσχιση της «Check_Power(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 84)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Check_Power (a_CP, b_CP)» σελίδες «53-54».

Κλήση της συνάρτησης "Power (A, B)" (γραμμή 87)

Στη γραμμή «87» η «Check_Power(...)» καλεί τη συνάρτηση «Power(…)», η οποία επιστρέφει το αποτέλεσμα της δύναμης «A^B», εφόσον δεν υπάρχει κάποιος περιορισμός ή μία τιμή σφάλματος (-1.0), εφόσον υπάρχει κάποιος περιορισμός. Το αποτέλεσμα καταχωρείται στη μεταβλητή «cp».

(~) Μη τιμή σφάλματος (γραμμές 88-92)
(~) Τιμή σφάλματος (γραμμές 93-97)

Στις γραμμές «88-97» υλοποιείται ένας έλεγχος με μία εντολή «if – else» για την τιμή που επιστρέφει η «Power(…)». Η συνθήκη είναι η «cp != -1.0» και άμα παράγει μία τιμή «True», δηλαδή, η «Power(…)» επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος, τότε θα εκτελεστούν οι εντολές της υποενότητας «Μη τιμή σφάλματος (γραμμές 88-92)». Διαφορετικά, αν η «Power(…)» επιστρέψει την τιμή «-1.0» (τιμή σφάλματος), τότε θα εκτελεστούν οι εντολές της υποενότητας «Τιμή σφάλματος (γραμμές 93-97)». Η διάσχιση της «if – else» έχει ως εξής :

(~) Μη τιμή σφάλματος (γραμμές 88-92)

Στις γραμμές «88-92» εκτελούνται οι εντολές της «if (cp != -1.0)» (γραμμή 88), για την περίπτωση που η «Power(…)» επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος. Η διάσχιση της «if» έχει ως εξής :

Εκτύπωση του αποτελέσματος της λειτουργίας [1] (A^B) (γραμμή 90)

Στη γραμμή «90» τυπώνεται στη «standard» έξοδο το περιεχόμενο της τιμής που επέστρεψε η «Power(…)» και καταχωρήθηκε στη μεταβλητή «cp», που είναι και το αποτέλεσμα της δύναμης «A^B».

(~) Τιμή σφάλματος (γραμμές 93-97)

Στις γραμμές «93-97» εκτελούνται οι εντολές της «else» (γραμμή 93) που αντιστοιχεί στην «if (cp != -1.0)» (γραμμή 78), για την περίπτωση που η «Power(…)» επιστρέφει μία τιμή που είναι είναι ίση με την τιμή σφάλματος. Προφανώς, πρόκειται για παραβίαση του περιορισμού και γι’ αυτό στη γραμμή «95» τυπώνεται στη «standard» έξοδο το χαρακτηριστικό μήνυμα.

Check_Valid_Power (A, B) **(γραμμές 100-113)**

Στις γραμμές «100-113» υλοποιείται η συνάρτηση “Check_Valid_Power(..)” (για την καταμέτρηση του πλήθους των έγκυρων λειτουργιών), τύπου «int» και με παραμέτρους τις μεταβλητές «a_CVP», «b_CVP» (τύπου «int»), όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» και «B», καλεί τη συνάρτηση «Power(…)» και ανάλογα με την τιμή που επέστρεψε, επιστρέφει εκεί που είναι ο έλεγχος του προγράμματος, μία σταθερά που δηλώνει ότι χρησιμοποιήθηκε έγκυρα ή άκυρα η λειτουργία [1]. Η διάσχιση της «Check_Valid_Power(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 102)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Check_Valid_Power (a_CVP, b_CVP)» σελίδα «54».

Κλήση της συνάρτησης "Power (A, B)" (γραμμή 104)

Στη γραμμή «104» η «Check_Valid_Power(…)» καλεί τη συνάρτηση «Power(…)», η οποία επιστρέφει το αποτέλεσμα της δύναμης «A^B», εφόσον δεν υπάρχει κάποιος περιορισμός ή μία τιμή σφάλματος (-1.0), εφόσον υπάρχει κάποιος περιορισμός. Το αποτέλεσμα καταχωρείται στη μεταβλητή «cvp».

(~) Μη τιμή σφάλματος (γραμμές 105-108)
(~) Τιμή σφάλματος (γραμμές 109-112)

Στις γραμμές «105-112» υλοποιείται ένας έλεγχος με μία εντολή «if – else» για την τιμή που επιστρέφει η «Power(…)». Η συνθήκη είναι η «cvp != -1.0» και άμα παράγει μία τιμή «True», δηλαδή, η «Power(…)» επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος, τότε θα εκτελεστούν οι εντολές της υποενότητας «Μη τιμή σφάλματος (γραμμές 105-108)». Διαφορετικά, αν η «Power(…)» επιστρέψει την τιμή «-1.0» (τιμή σφάλματος), τότε θα εκτελεστούν οι εντολές της υποενότητας «Τιμή σφάλματος (γραμμές 109-112)». Η διάσχιση της «if – else» έχει ως εξής :

(~) Μη τιμή σφάλματος (γραμμές 105-108)

Στις γραμμές «105-108» εκτελούνται οι εντολές της «if (cvp != -1.0)» (γραμμή 105), για την περίπτωση που η «Power(…)» επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος. Η διάσχιση της «if» έχει ως εξής :

Επιστροφή τιμής έγκυρης λειτουργίας (γραμμή 107)

Στη γραμμή «107» η εντολή «return 1» επιστρέφει την ακέραια τιμή «1» εκεί που είναι ο έλεγχος του προγράμματος, δηλώνοντας ότι η λειτουργία [1]

χρησιμοποιήθηκε έγκυρα.

(~) Τιμή σφάλματος (γραμμές 109-112)

Στις γραμμές «98-101» εκτελούνται οι εντολές της «else» (γραμμή 98) που αντιστοιχεί στην «if (cvp != -1.0)» (γραμμή 94), για την περίπτωση που η «Power(…)» επιστρέφει μία τιμή που είναι ίση με την τιμή σφάλματος. Η διάσχιση της «else» έχει ως εξής :

Επιστροφή τιμής άκυρης λειτουργίας (γραμμή 111)

Στη γραμμή «111» η εντολή «return 0» επιστρέφει την ακέραια τιμή «0» εκεί που είναι ο έλεγχος του προγράμματος, δηλώνοντας ότι η λειτουργία [1] εκτελέστηκε άκυρα.

Factorial (A_B) (γραμμές 115-134)

Στις γραμμές «115-134» υλοποιείται η συνάρτηση «Factorial(…)», τύπου «int» και με παραμέτρους τη μεταβλητή «a_b_F» (τύπου «int»), όπου δέχεται για είσοδο έναν από τους δύο ακέραιους αριθμούς «A» και «B», υπολογίζει το παραγοντικό «A!» ή «B!» και επιστρέφει το αποτέλεσμα ή μία τιμή σφάλματος, εφόσον, υπάρχει κάποιος περιορισμός. Η διάσχιση της «Factorial(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμές 117-119)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Factorial (int a_b_F)» σελίδα «54»

Αρχικοποίηση μεταβλητών (γραμμές 118, 119)

Στις γραμμές «118» και «119» γίνεται μία αρχικοποίηση της μεταβλητής «error_F» (τιμή σφάλματος), όπου καταχωρείται η τιμή «-1» (γραμμή 119) και της μεταβλητής «p» (το παραγοντικό), όπου καταχωρείται η τιμή «1» (γραμμή 118).

(~) A_B >= 0 (γραμμές 121-129)
(~) A_B < 0 (γραμμές 130-133)

Στις γραμμές «121-133» υλοποιείται ένας έλεγχος με την εντολή «if – else», όσον αφορά τον περιορισμό που δεν καθιστά εφικτό τον υπολογισμό του παραγοντικού «A!» και «B!». Η συνθήκη είναι η «a_b_F >= 0» και άμα παράγει μία τιμή «True», τότε θα εκτελεστούν οι εντολές της υποενότητας «A_B >= 0 (γραμμές 121-129)». Αντιθέτως, θα εκτελεστούν οι εντολές της υποενότητας «A_B < 0 (γραμμές 130-133)». Η διάσχιση του ελέγχου έχει ως εξής :

(~) A_B >= 0 (γραμμές 121-129)

Στις γραμμές «121-129» εκτελούνται οι εντολές της «if (a_b_F >= 0)» (γραμμή 121), για την περίπτωση που ο «A» ή ο «B» περιέχουν τιμή μεγαλύτερη ή ίση του «0». Η διάσχιση της «if» έχει ως εξής :

Βρόχος (γραμμές 123-126)

Στις γραμμές «123-126» υλοποιείται ένας βρόχος με την εντολή «for» για τον υπολογισμό του παραγοντικού. Ο βρόχος εκτελείται για όσο η βοηθητική μεταβλητή «i» που τον ελέγχει με αρχική τιμή, τη τιμή «1» και αυξάνοντας κατά «1» κάθε φορά που εκτελείται ο βρόχος με την παράσταση «i++», περιέχει τιμή μεγαλύτερη του ακεραίου αριθμού που δέχεται για είσοδο η «Factorial(...)» και καταχωρείται στη μεταβλητή «a_b_F». Η διάσχιση του βρόχου έχεις ως εξής :

Υπολογισμός της τιμής του παραγοντικού του "A!" ή του "B!" (γραμμή 125)

Στη γραμμή «125» υπολογίζεται το παραγοντικό «A!» ή «B!» που εκτελείται σε κάθε επανάληψη του βρόχου με τη παράσταση «p = p * i». Το αποτέλεσμα καταχωρείται στη μεταβλητή «p», όπου, έχει για αρχική τιμή τη τιμή «1» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 118)».

Επιστροφή της τιμής του "A!" ή του "B!" (γραμμή 128)

Στη γραμμή «128» η εντολή «return power» επιστρέφει το περιεχόμενο της «power», δηλαδή, το αποτέλεσμα του παραγοντικού «A!» ή «B!», εκεί που είναι ο έλεγχος του προγράμματος.

(~) A_B < 0 (γραμμές 130-133)

Στις γραμμές «130-133» εκτελούνται οι εντολές της «else» (γραμμή 130) που αντιστοιχεί στην «if (a_b_F >= 0)» (γραμμή 130), για την περίπτωση που ο «A» ή ο «B» δεν περιέχουν τιμή μεγαλύτερη ή ίση του «0». Η διάσχιση της «else» έχει ως εξής :

Επιστροφή τιμής σφάλματος (γραμμή 132)

Στη γραμμή «132» η εντολή «return error_F» επιστρέφει το περιεχόμενο της «error_F» (τιμή σφάλματος), δηλαδή, την τιμή «-1» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 119)»), εκεί που είναι ο έλεγχος του προγράμματος.

Check_Factorial_A (A) (γραμμές 136-149)

Στις γραμμές «136-149» υλοποιείται η συνάρτηση «Check_Factorial_A(...)», τύπου «void» και με παράμετρο τη μεταβλητή «a_CFA» (τύπου «int»), όπου δέχεται για είσοδο τον πρώτο ακέραιο αριθμό «A», καλεί τη συνάρτηση «Factorial(…)» με παράμετρο τον «A» και ανάλογα με την τιμή που επέστρεψε

τυπώνει στη «standard» έξοδο τα χαρακτηριστικά μηνύματα. Η διάσχιση της «Check_Factorial_A(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 138)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Check_Factorial_A (a_CFA)» σελίδα «54».

Κλήση της συνάρτησης " Factorial (A)" (γραμμή 140)

Στη γραμμή «140» η «Check_Factorial_A(...)» καλεί τη συνάρτηση «Factorial(…)» με παράμετρο τη μεταβλητή «a_CFA» (ο πρώτος ακέραιος αριθμός «A»), η οποία επιστρέφει το αποτέλεσμα του παραγοντικού «A!», εφόσον δεν υπάρχει κάποιος περιορισμός ή μία τιμή σφάλματος (-1), εφόσον υπάρχει κάποιος περιορισμός. Το αποτέλεσμα καταχωρείται στη μεταβλητή «cf_a».

(~) Μη τιμή σφάλματος (γραμμές 141-144)
(~) Τιμή σφάλματος (γραμμές 145-148)

Στις γραμμές «141-148» υλοποιείται ένας έλεγχος με μία εντολή «if – else» για την τιμή που επιστρέφει η «Factorial(…)» με παράμετρο τη μεταβλητή «a_CFA» (ο πρώτος ακέραιος αριθμός «A»). Η συνθήκη είναι η «cf_a != -1» και άμα παράγει μία τιμή «True», δηλαδή, η «Factorial(...)» με παράμετρο τον ακέραιο αριθμό «A» (a_CFA), επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος, τότε θα εκτελεστούν οι εντολές της υποενότητας «Μη τιμή σφάλματος (γραμμές 141-144)». Διαφορετικά, αν η «Factorial(…)» με παράμετρο τον ακέραιο αριθμός «A» (a_CFA) επιστρέψει την τιμή «-1» (τιμή σφάλματος), τότε θα εκτελεστούν οι εντολές της υποενότητας «Τιμή σφάλματος (γραμμές 145-148)». Η διάσχιση της «if – else» έχει ως εξής :

(~) Μη τιμή σφάλματος (γραμμές 141-144)

Στις γραμμές «141-144» εκτελούνται οι εντολές της «if (cf_a != -1)» (γραμμή 141), για την περίπτωση που η «Factorial(…)» με παράμετρο τον ακέραιο αριθμό «A» (a_CFA), επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος. Η διάσχιση της «if» έχει ως εξής :

Εκτύπωση του αποτελέσματος της υπολειτουργίας [2] "A!" (γραμμή 143)

Στη γραμμή «143» τυπώνεται στη «standard» έξοδο το περιεχόμενο της τιμής που επέστρεψε η «Factorial(…)» με παράμετρο τον ακέραιο αριθμό «A» (a_CFA) και καταχωρήθηκε στη μεταβλητή «cf_a», που είναι και το αποτέλεσμα του παραγοντικού «A!». Αξίζει να σημειωθεί ότι το αλφαριθμητικό μορφοποίησης είναι το «%20d», δηλαδή, το αποτέλεσμα θα τυπωθεί αμεσώς μετά από «20» κενούς χαρακτήρες. Αυτό αποσκοπεί στην ομοιόμορφη στοίχιση

των αποτελεσμάτων.

(~) Τιμή σφάλματος (γραμμές 145-148)

Στις γραμμές «145-148» εκτελούνται οι εντολές της «else» (γραμμή 145) που αντιστοιχεί στην «if (cf_a != -1)» (γραμμή 141), για την περίπτωση που η η «Factorial(…)» με παράμετρο τον ακέραιο αριθμό «A» (a_CFA), επιστρέφει μία τιμή που είναι είναι ίση με την τιμή σφάλματος. Προφανώς, πρόκειται για παραβίαση του περιορισμού και γι' αυτό στη γραμμή «147» τυπώνεται στη «standard» έξοδο το χαρακτηριστικό μήνυμα.

Check Factorial B (B) (γραμμές 151-166)

Στις γραμμές «151-166» υλοποιείται η συνάρτηση «Check_Factorial_B(...)», τύπου «void» και με παράμετρο τη μεταβλητή «b_CFB» (τύπου «int»), όπου δέχεται για είσοδο τον δεύτερο ακέραιο αριθμό «B», καλεί τη συνάρτηση «Factorial(…)» με παράμετρο τον «B» και ανάλογα με την τιμή που επέστρεψε, τυπώνει στη «standard» έξοδο τα χαρακτηριστικά μηνύματα. Η διάσχιση της «Check_Factorial_B(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 153)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Check_Factorial_B (b_CFB)», σελίδες«54-55».

Κλήση της συνάρτησης " Factorial (B)" (γραμμή 155)

Στη γραμμή «155» η «Check_Factorial_B(...)» καλεί τη συνάρτηση «Factorial(…)» με παράμετρο τη μεταβλητή «b_CFB» (ο δεύτερος ακέραιος αριθμός «B»), η οποία επιστρέφει το αποτέλεσμα του παραγοντικού «B!», εφόσον δεν υπάρχει κάποιος περιορισμός ή μία τιμή σφάλματος (-1), εφόσον υπάρχει κάποιος περιορισμός. Το αποτέλεσμα καταχωρείται στη μεταβλητή «cf_b».

(~) Μη τιμή σφάλματος (γραμμές 156-160)
(~) Τιμή σφάλματος (γραμμές 161-165)

Στις γραμμές «156-165» υλοποιείται ένας έλεγχος με μία εντολή «if – else» για την τιμή που επιστρέφει η «Factorial(…)» με παράμετρο τη μεταβλητή «b_CFB» (ο δεύτερος ακέραιος αριθμός «B»). Η συνθήκη είναι η «cf_b != -1» και άμα παράγει μία τιμή «True», δηλαδή, η «Factorial(…)» με παράμετρο τον ακέραιο αριθμό «B» (b_CFB), επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος, τότε θα εκτελεστούν οι εντολές της υποενότητας «Μη τιμή σφάλματος (γραμμές 156-160)». Διαφορετικά, αν η «Factorial(…)» με παράμετρο τον ακέραιο αριθμό «B» (b_CFB) επιστρέψει την τιμή «-1» (τιμή σφάλματος), τότε θα εκτελεστούν οι εντολές της υποενότητας «Τιμή σφάλματος

(γραμμές 161-165)». Η διάσχιση της «if – else» έχει ως εξής :

(~) Μη τιμή σφάλματος (γραμμές 156-160)

Στις γραμμές «156-160» εκτελούνται οι εντολές της «if (cf_b != -1)» (γραμμή 156), για την περίπτωση που η «Factorial(…)» με παράμετρο τον ακέραιο αριθμό «B» (b_CFB), επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος. Η διάσχιση της «if» έχει ως εξής :

Εκτύπωση του αποτελέσματος της υπολειτουργίας [2] "B!" (γραμμή 158)

Στη γραμμή «158» τυπώνεται στη «standard» έξοδο το περιεχόμενο της τιμής που επέστρεψε η «Factorial(…)» με παράμετρο τον ακέραιο αριθμό «B» (b_CFB) και καταχωρήθηκε στη μεταβλητή «cf_b», που είναι και το αποτέλεσμα του παραγοντικού «B!». Αξίζει να σημειωθεί ότι το αλφαριθμητικό μορφοποίησης είναι το «%20d», δηλαδή, το αποτέλεσμα θα τυπωθεί αμεσώς μετά από «20» κενούς χαρακτήρες. Αυτό αποσκοπεί στην ομοιόμορφη στοίχιση των αποτελεσμάτων.

(~) Τιμή σφάλματος (γραμμές 161-165)

Στις γραμμές «161-165» εκτελούνται οι εντολές της «else» (γραμμή 161) που αντιστοιχεί στην «if (cf_a != -1)» (γραμμή 141), για την περίπτωση που η η «Factorial(…)» με παράμετρο τον ακέραιο αριθμό «B» (b_CFB), επιστρέφει μία τιμή που είναι είναι ίση με την τιμή σφάλματος. Προφανώς, πρόκειται για παραβίαση του περιορισμού και γι' αυτό στη γραμμή «163» τυπώνεται στη «standard» έξοδο το χαρακτηριστικό μήνυμα.

Check_Factorial (A, B) (γραμμές 168-175)

Στις γραμμές «168-175» υλοποιείται η συνάρτηση «Check_Factorial(...)», τύπου «void», όπου δέχεται για είσοδο τον πρώτο ακέραιο αριθμό «A» (a_CF) και τον δεύτερο ακέραιο αριθμό «B» (b_CF), καλεί τις συναρτήσεις «Check_Factorial_A(…)» και «Check_Factorial_B(…)» και τυπώνει στη «standard» έξοδο το χαρακτηριστικό μήνυμα για την λειτουργία «2» (γραμμή 172). Η διάσχιση της «Check_Factorial(…)» έχει ως εξής :

Κλήση της συνάρτησης "Check_Factorial_A (A)" (γραμμή 173)

Στη γραμμή «173» η «Check_Factorial(…)» καλεί τη συνάρτηση «Check_Factorial_A(…)» με παράμετρο τον πρώτο ακέραιο αριθμό «A» (a_CF) που υλοποιείται στις γραμμές «136-149».

Κλήση της συνάρτησης "Check_Factorial_B (B)" (γραμμή 174)

Στη γραμμή «174» η «Check_Factorial(…)» καλεί τη συνάρτηση

«Check_Factorial_B(…)» με παράμετρο τον δεύτερο ακέραιο αριθμό «B» (b_CF) που υλοποιείται στις γραμμές «151-166»

Check_Valid_Factorial (A, B) (γραμμές 177-191)

Στις γραμμές «177-191» υλοποιείται η συνάρτηση “Check_Valid_Factorial”, (για την καταμέτρηση του πλήθους των έγκυρων λειτουργιών) τύπου «int» και με παραμέτρους τις μεταβλητές «a_CVP», «b_CVP» (τύπου «int»), όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» και «B», καλεί τη συνάρτηση «Factorial(…)» δύο φορές (την μία με παράμετρο τον ακέραιο αριθμό «A» και την άλλη με τον ακέραιο αριθμό «B») και ανάλογα με τις τιμές που επέστρεψαν, επιστρέφει εκεί που είναι ο έλεγχος του προγράμματος, μία σταθερά που δηλώνει ότι χρησιμοποιήθηκε έγκυρα ή άκυρα η λειτουργία [2]. Η διάσχιση της «Check_Valid_Factorial(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 179)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Check_Valid_Factorial (a_CVF, b_CVF)» σελίδα «55».

Κλήση της συνάρτησης "Factorial (A)" (γραμμή 181)

Στη γραμμή «181» η «Check_Valid_Factorial(…)» καλεί τη συνάρτηση «Factorial(…)» με παράμετρο τον πρώτο ακέραιο αριθμό «A» (a_CVF), η οποία επιστρέφει το αποτέλεσμα της δύναμης «A!», εφόσον δεν υπάρχει κάποιος περιορισμός ή μία τιμή σφάλματος (-1), εφόσον υπάρχει κάποιος περιορισμός. Το αποτέλεσμα καταχωρείται στη μεταβλητή «cvf_a».

Κλήση της συνάρτησης "Factorial (B)" (γραμμή 182)

Στη γραμμή «182» η «Check_Valid_Factorial(…)» καλεί τη συνάρτηση «Factorial(…)» με παράμετρο τον δεύτερο ακέραιο αριθμό «B» (b_CVF), η οποία επιστρέφει το αποτέλεσμα της δύναμης «B!», εφόσον δεν υπάρχει κάποιος περιορισμός ή μία τιμή σφάλματος (-1), εφόσον υπάρχει κάποιος περιορισμός. Το αποτέλεσμα καταχωρείται στη μεταβλητή «cvf_b».

(~) Μη τιμή σφάλματος (γραμμές 183-186)
(~) Τιμή σφάλματος (γραμμές 187-190)

Στις γραμμές «183-190» υλοποιείται ένας έλεγχος με μία εντολή «if – else» για τις τιμές που επιστρέφει η «Factorial(…)», μία με παράμετρο τον πρώτο ακέραιο αριθμό «A» και μία με παράμετρο τον δεύτερο ακέραιο αριθμό «B». Η συνθήκη είναι η «cvf_a != -1 && cvf_b != -1» και άμα παράγει μία τιμή «True», δηλαδή, η «Factorial(…)» επιστρέφει, μία με παράμετρο τον πρώτο ακέραιο αριθμό «A» και μία με παράμετρο τον δεύτερο ακέραιο αριθμό «B», τιμές που δεν είναι ίσες με την τιμή σφάλματος, τότε, θα εκτελεστούν οι εντολές της

υποενότητας «Μη τιμή σφάλματος (γραμμές 183-186)». Διαφορετικά, αν η «Factorial(…)» επιστρέψει την τιμή «-1.0» (τιμή σφάλματος) και στις δύο περιπτώσεις, τότε θα εκτελεστούν οι εντολές της υποενότητας «Τιμή σφάλματος (γραμμές 187-190)». Η διάσχιση της «if – else» έχει ως εξής :

(~) Μη τιμή σφάλματος (γραμμές 183-186)

Στις γραμμές «183-186» εκτελούνται οι εντολές της «if (cvf_a != -1 && cvf_b != -1)» (γραμμή 183), για την περίπτωση που η «Factorial(…)» επιστρέφει μία τιμή, μία με παράμετρο τον πρώτο ακέραιο αριθμό «A» και μία με παράμετρο τον δεύτερο ακέραιο αριθμό «B», που δεν είναι ίση με την τιμή σφάλματος. Η διάσχιση της «if» έχει ως εξής :

Επιστροφή τιμής έγκυρης λειτουργίας (γραμμή 185)

Στη γραμμή «185» η εντολή «return 1» επιστρέφει την ακέραια τιμή «1» εκεί που είναι ο έλεγχος του προγράμματος, δηλώνοντας ότι η λειτουργία [2] χρησιμοποιήθηκε έγκυρα.

(~) Τιμή σφάλματος (γραμμές 187-190)

Στις γραμμές «187-190» εκτελούνται οι εντολές της «else» (γραμμή 187) που αντιστοιχεί στην «if (cvf_a != -1 && cvf_b != -1)» (γραμμή 94), για την περίπτωση που η «Power(…)» επιστρέφει μία τιμή, μία με παράμετρο τον πρώτο ακέραιο αριθμό «A» και μία με παράμετρο τον δεύτερο ακέραιο αριθμό «B» που είναι ίση με την τιμή σφάλματος. Η διάσχιση της «else» έχει ως εξής :

Επιστροφή τιμής άκυρης λειτουργίας (γραμμή 189)

Στη γραμμή «189» η εντολή «return 0» επιστρέφει την ακέραια τιμή «0» εκεί που είναι ο έλεγχος του προγράμματος, δηλώνοντας ότι η λειτουργία [2] εκτελέστηκε άκυρα.

Combinations (A, B) (γραμμές 193-211)

Στις γραμμές «193-211» υλοποιείται η συνάρτηση «Combinations(…)», τύπου «int» και με παραμέτρους τις μεταβλητές «a_C» και «b_C» (τύπου «int»), όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» και «B», υπολογίζει το πλήθος των συνδυασμών «A» ανά «B» και επιστρέφει το αποτέλεσμα ή μία τιμή σφάλματος, εφόσον, υπάρχει κάποιος περιορισμός. Η διάσχιση της «Combinations(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμές 195-196)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Combinations (int a_C, int b_C)» σελίδες «55-56».

Αρχικοποίηση μεταβλητών (γραμμή 196)

Στη γραμμή «196» γίνεται μία αρχικοποίηση της μεταβλητής «error_C» (τιμή σφάλματος), όπου καταχωρείται η τιμή «-1».

(~) A > B ΚΑΙ A >= 0 ΚΑΙ B >= 0 (γραμμές 198-206)
(~) A <= B Ή A < 0 Ή B < 0 (γραμμές 207-210)

Στις γραμμές «198-210» υλοποιείται ένας έλεγχος με την εντολή «if – else», όσον αφορά τον περιορισμό που δεν καθιστά εφικτό τον υπολογισμό του πλήθους των συνδυασμών «A» άνα «B». Η συνθήκη είναι η «a_C > b_C && a_C >= 0 && b_C >= 0» και άμα παράγει μία τιμή «True», τότε θα εκτελεστούν οι εντολές της υποενότητας «A > B ΚΑΙ A >= 0 ΚΑΙ B >= 0 (γραμμές 198-206)». Αντιθέτως, θα εκτελεστούν οι εντολές της υποενότητας «A <= B Ή A < 0 Ή B < 0 (γραμμές 207-210)». Η διάσχιση του ελέγχου έχει ως εξής :

(~) A > B ΚΑΙ A >= 0 ΚΑΙ B >= 0 (γραμμές 198-206)

__Στις γραμμές «198-206» εκτελούνται οι εντολές της «if (a_C > b_C && a_C >= 0 && b_C >= 0)» (γραμμή 198), για την περίπτωση που ο «A» και ο «B» περιέχουν τιμή μεγαλύτερη ή ίση του «0» και ο «A» περιέχει τιμή μεγαλύτερη από αυτή του «B». Η διάσχιση της «if» έχει ως εξής :

Κλήση της συνάρτησης "Factorial (A)" (γραμμή 200)

Στη γραμμή «200» η «Combinations(…)» καλεί τη συνάρτηση «Factorial(…)» με παράμετρο τον πρώτο ακέραιο αριθμό «A» (a_C) που υλοποιείται στις γραμμές «115-134». Η τιμή που επιστρέφει, καταχωρείται στη μεταβλητή «i».

Κλήση της συνάρτησης "Factorial (A)" (γραμμή 201)

Στη γραμμή «201» η «Combinations(…)» καλεί τη συνάρτηση «Factorial(…)» με παράμετρο τον δεύτερο ακέραιο αριθμό «B» (b_C) που υλοποιείται στις γραμμές «115-134». Η τιμή που επιστρέφει, καταχωρείται στη μεταβλητή «j».

Κλήση της συνάρτησης "Factorial (A - B)" (γραμμή 202)

Στη γραμμή «202» η «Combinations(…)» καλεί τη συνάρτηση «Factorial(…)» με παράμετρο τη διαφορά του πρώτου ακέραιου αριθμού «A» (a_C) με τον δεύτερο ακέραιο αριθμό «B» (b_C), που υλοποιείται στις γραμμές «115-134». Η τιμή που επιστρέφει, καταχωρείται στη μεταβλητή «k».

Υπολογισμός του πλήθους των συνδυασμών "A" ανά "B" (γραμμή 203)

Στη γραμμή «203» υπολογίζεται με την αριθμητική παράσταση «l / (j * k)» το πλήθος των συνδυασμών «A» άνα «B» και το αποτέλεσμα καταχωρείται στη μεταβλητή «combos».

Επιστροφή του πλήθους των συνδυασμών "A" ανά "B" (γραμμή 205)

Στη γραμμή «205» η εντολή «return combos» επιστρέφει το περιεχόμενο της μεταβλητής «combos» (το πλήθος των συνδυασμών «A» ανά «B»), εκεί που είναι ο έλεγχος του προγράμματος.

(~) A <= B Ή A < 0 Ή B < 0 (γραμμές 207-210)

Στις γραμμές «207-210» εκτελούνται οι εντολές της «else» (γραμμή 207) που αντιστοιχούν στην «if (a_C > b_C && a_C >= 0 && b_C >= 0)» (γραμμή 198), για την περίπτωση που ο «A» ή ο «B» περιέχουν τιμή μικρότερη του «0» ή ο «A» περιέχει τιμή μικρότερη από αυτή του «B». Η διάσχιση της «if» έχει ως εξής:

Επιστροφή τιμής σφάλματος (γραμμή 209)

Στη γραμμή «209» η εντολή «return error_C» επιστρέφει το περιεχόμενο της μεταβλητής «error_C» (τιμή σφάλματος), δηλαδή, την τιμή «-1» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 196)»), εκεί που είναι ο έλεγχος του προγράμματος.

Check_Combinations (A, B) (γραμμές 213-231)

Στις γραμμές «213-231» υλοποιείται η συνάρτηση «Check_Combinations», τύπου «void» και με παραμέτρους τις μεταβλητές «a_CC», «b_CC» (τύπου «int»), όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» και «B», καλεί τη συνάρτηση «Combinations(…)» και ανάλογα με την τιμή που επέστρεψε τυπώνει στη «standard» έξοδο τα χαρακτηριστικά μηνύματα. Στη γραμμή «219» τυπώνεται στη «standard» έξοδο το χαρακτηριστικό μήνυμα για την λειτουργία «3». Η διάσχιση της «Check_Combinations(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμή 217)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Check_Combinations (a_CC, b_CC)», σελίδα «56».

Κλήση της συνάρτησης "Combinations (A, B)" (γραμμή 220)

Στη γραμμή «220» η «Check_Combinations(...)» καλεί τη συνάρτηση «Combinations(…)», η οποία επιστρέφει το αποτέλεσμα του πλήθους των

συνδυασμών «A» ανά «B», εφόσον δεν υπάρχει κάποιος περιορισμός ή μία τιμή σφάλματος (-1), εφόσον υπάρχει κάποιος περιορισμός. Το αποτέλεσμα καταχωρείται στη μεταβλητή «cc».

(~) Μη τιμή σφάλματος (γραμμές 221-225)
(~) Τιμή σφάλματος (γραμμές 226-230)

Στις γραμμές «221-225» υλοποιείται ένας έλεγχος με μία εντολή «if – else» για την τιμή που επιστρέφει η «Combinations(…)». Η συνθήκη είναι η «cc != -1» και άμα παράγει μία τιμή «True», δηλαδή, η «Combinations(…)» επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος, τότε θα εκτελεστούν οι εντολές της υποενότητας «Μη τιμή σφάλματος (γραμμές 221-225)». Διαφορετικά, αν η «Combinations(…)» επιστρέψει την τιμή «-1» (τιμή σφάλματος), τότε θα εκτελεστούν οι εντολές της υποενότητας «Τιμή σφάλματος (γραμμές 226-230)». Η διάσχιση της «if – else» έχει ως εξής :

(~) Μη τιμή σφάλματος (γραμμές 221-225)

Στις γραμμές «221-225» εκτελούνται οι εντολές της «if (cc != -1)» (γραμμή 221), για την περίπτωση που η «Combinations(…)» επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος. Η διάσχιση της «if» έχει ως εξής :

Εκτύπωση του αποτελέσματος της λειτουργίας [3] (A! / B! * (A - B)!))
(γραμμή 223)

Στη γραμμή «223» τυπώνεται στη «standard» έξοδο το περιεχόμενο της τιμής που επέστρεψε η «Combinations(…)» και καταχωρήθηκε στη μεταβλητή «cc», που είναι και το αποτέλεσμα της παράστασης «A! / B! * (A – B)!» (το πλήθος των συνδυασμών «A» ανά «B».

(~) Τιμή σφάλματος (γραμμές 226-230)

Στις γραμμές «226-230» εκτελούνται οι εντολές της «else» (γραμμή 226) που αντιστοιχεί στην «if (cc != -1)» (γραμμή 221), για την περίπτωση που η «Combinations(…)» επιστρέφει μία τιμή που είναι είναι ίση με την τιμή σφάλματος. Προφανώς, πρόκειται για παραβίαση του περιορισμού και γι' αυτό στη γραμμή «228» τυπώνεται στη «standard» έξοδο το χαρακτηριστικό μήνυμα.

Check_Valid_Combinations (A, B) (γραμμές 233-246)

Στις γραμμές «223-246» υλοποιείται το αυτόνομο υποπρόγραμμα "Check_Valid_Combinations(…) για την καταμέτρηση του πλήθους των έγκυρων λειτουργιών, τύπου «int» και με παραμέτρους τις μεταβλητές «a_CVC», «b_CVC» (τύπου «int»), όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» και «B», καλεί τη συνάρτηση «Combinations(…)» και

ανάλογα με την τιμή που επέστρεψε, επιστρέφει εκεί που είναι ο έλεγχος του προγράμματος, μία σταθερά που δηλώνει ότι χρησιμοποιήθηκε έγκυρα ή άκυρα η λειτουργία [3]. Η διάσχιση της «Check_Valid_Combinations(…)» έχει ως εξής:

Δήλωση μεταβλητών (γραμμή 235)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Check_Valid_Combinations (a_CVC, b_CVC)» σελίδα «56».

Κλήση της συνάρτησης "Combinations (A, B)" (γραμμή 237)

Στη γραμμή «237» η «Check_Valid_Combinations(…)» καλεί τη συνάρτηση «Combinations(…)», η οποία επιστρέφει το αποτέλεσμα του πλήθους των συνδυασμών «A» ανά «B», εφόσον δεν υπάρχει κάποιος περιορισμός ή μία τιμή σφάλματος (-1), εφόσον υπάρχει κάποιος περιορισμός. Το αποτέλεσμα καταχωρείται στη μεταβλητή «cvc».

(~) Μη τιμή σφάλματος (γραμμές 238-241)
(~) Τιμή σφάλματος (γραμμές 242-245)

Στις γραμμές «238-245» υλοποιείται ένας έλεγχος με μία εντολή «if – else» για την τιμή που επιστρέφει η «Combinations(…)». Η συνθήκη είναι η «cvc != -1» και άμα παράγει μία τιμή «True», δηλαδή, η «Combinations(…)» επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος, τότε θα εκτελεστούν οι εντολές της υποενότητας «Μη τιμή σφάλματος (γραμμές 238-241)». Διαφορετικά, αν η «Combinations(…)» επιστρέψει την τιμή «-1» (τιμή σφάλματος), τότε θα εκτελεστούν οι εντολές της υποενότητας «Τιμή σφάλματος (γραμμές 242-245)». Η διάσχιση της «if – else» έχει ως εξής :

(~) Μη τιμή σφάλματος (γραμμές 238-241)

Στις γραμμές «238-241» εκτελούνται οι εντολές της «if (cvc != -1)» (γραμμή 238), για την περίπτωση που η «Combinations(…)» επιστρέφει μία τιμή που δεν είναι ίση με την τιμή σφάλματος. Η διάσχιση της «if» έχει ως εξής :

Επιστροφή τιμής έγκυρης λειτουργίας (γραμμή 240)

Στη γραμμή «240» η εντολή «return 1» επιστρέφει την ακέραια τιμή «1» εκεί που είναι ο έλεγχος του προγράμματος, δηλώνοντας ότι η λειτουργία [3] εκτελέστηκε έγκυρα.

(~) Τιμή σφάλματος (γραμμές 242-245)

Στις γραμμές «242-245» εκτελούνται οι εντολές της «else» (γραμμή 242) που αντιστοιχεί στην «if (cvc != -1)» (γραμμή 238), για την περίπτωση που η «Combinations(…)» επιστρέφει μία τιμή που είναι ίση με την τιμή σφάλματος.

Η διάσχιση της «else» έχει ως εξής :

Επιστροφή τιμής άκυρης λειτουργίας (γραμμή 244)

Στη γραμμή «244» η εντολή «return 0» επιστρέφει την ακέραια τιμή «0» εκεί που είναι ο έλεγχος του προγράμματος, δηλώνοντας ότι η λειτουργία [3] εκτελέστηκε άκυρα.

Exit () (γραμμές 226-230)

Στις γραμμές «248-253» υλοποιείται η συνάρτηση «Exit(…)» (για την καταμέτρηση του πλήθους των έγκυρων λειτουργιών), τύπου «int» (χωρίς παραμέτρους), όπου επιστρέφει την τιμή «1» κάθε φορά που καλείται στο πρόγραμμα. Η διάσχιση της «Exit(…)» έχει ως εξής :

Δήλωση και αρχικοποίηση μεταβλητών (γραμμή 250)

Στη γραμμή «250» εκχωρείται η αρχική τιμή «1» στη μεταβλητή «cnt» (βοηθητική μεταβλητή για την καταμέτρηση της έγκυρης λειτουργίας [4]).

Επιστροφή τιμής έγκυρης λειτουργίας [4] (γραμμή 252)

Στη γραμμή «252» η εντολή «return cnt» επιστρέφει το περιεχόμενο της «cnt», δηλαδή, την ακέραια τιμή «1» (βλ. «Δήλωση και αρχικοποίηση μεταβλητών (γραμμή 250)»), εκεί που είναι ο έλεγχος του προγράμματος, δηλώνοντας ότι η λειτουργία [4] εκτελέστηκε έγκυρα.

Menu (A, B) (γραμμές 255-297)

Στις γραμμές «255-297» υλοποιείται η συνάρτηση «Menu(…)», τύπου «void» και με παράμετρους τις μεταβλητές «a_M» και «b_M», δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» και «B». Η συνάρτηση διαβάζει επανειλημμένα από τη «standard» είσοδο έναν ακέραιο αριθμό που αντιπροσωπεί μία τις «4» λειτουργίες, καλεί την αντίστοιχη συνάρτηση που εκτελεί αναλυτικά την λειτουργία που διαβάστηκε και τυπώνει στη «standard» έξοδο το πλήθος των έγκυρων λειτουργιών μετά το τέλος της επανάληψης. Η διάσχιση της «Menu(…)» έχει ως εξής :

Δήλωση μεταβλητών (γραμμές 259-263)

Βλ. ενότητα «Μεταβλητές», υποενότητα «Menu (a_M, b_M)», σελίδες «16-17»

Αρχικοποίηση μεταβλητών (γραμμές 260, 261, 262, 263)

Στις γραμμές «260», «261», «262» και «263» γίνεται μία αρχικοποίηση μεταβλητών. Πιο αναλυτικά, στη γραμμή «260» εκχωρείται η τιμή «0» στη

μεταβλητή «m_P» (το πλήθος των έγκυρων λειτουργιών [1]), στη γραμμή «261» εκχωρείται η τιμή «0» στη μεταβλητή «m_F» (το πλήθος των έγκυρων λειτουργιών [2]), στη γραμμή «262» εκχωρείται η τιμή «0» στη μεταβλητή «m_C» (το πλήθος των έγκυρων λειτουργιών [3]) και στη γραμμή «263» εκχωρείται η τιμή «0» στη μεταβλητή «m_E» (το πλήθος των έγκυρων λειτουργιών [4]).

Βρόχος (γραμμές 265-290)

Στις γραμμές «265-290» πραγματοποιείται ένας βρόχος με την εντολή επανάληψης «do – while» με συνθήκη τερματισμού του βρόχου, τη παράσταση «ch != 4» (ο αριθμός λειτουργίας που διαβάστηκε). Ο βρόχος εκτελείται τουλάχιστον μία φορά και μετά για όσο η παράσταση αυτή παράγει αποτέλεσμα μία τιμή «True» (για όσο ο χρήστης δεν επιλέγει την λειτουργία [4] της εξόδου). Η διάσχιση του βρόχου έχει ως εξής :

Το μενού με τις επιλογές λειτουργιών (γραμμές 267, 268, 269, 270)

Στις γραμμές «267», «268», «269» και «270» τυπώνεται στη «standard» έξοδο το μενού με τις διαθέσιμες λειτουργίες που μπορεί να επιλέξει ο χρήστης.

Εισαγωγή επιλογής λειτουργίας (γραμμή 272)

Στη γραμμή «272» διαβάζεται από τη «standard» είσοδο με τη συνάρτηση «scanf()», ο αριθμός που αντιπροσωπεύει μία επιλογή λειτουργίας συνοδευόμενο με το κατάλληλο μήνυμα που τυπώνεται στη «standard» έξοδο με την συνάρτηση «printf()» (γραμμή 271) κάθε φορά που εκτελείται ο βρόχος.

(~) Έγκυρη επιλογή λειτουργίας (γραμμές 273-283)
(~) Άκυρη επιλογή λειτουργίας (γραμμές 285-288)

Στις γραμμές «273-288» υλοποιείται με μία εντολή ελέγχου «if-else» ένας έλεγχος για την περίπτωση εισαγωγής έγκυρης επιλογής λειτουργίας ή άκυρης.

Διεξοδικά, αν ο χρήστης εισάγει έγκυρη επιλογή λειτουργίας (ch >= 1 && ch <= 4), τότε θα εκτελεστούν οι εντολές της υποενότητας «Έγκυρη επιλογή λειτουργίας (γραμμές 273-283)», διαφορετικά αν εισάγει άκυρη επιλογή λειτουργίας, τότε θα εκτελεστούν οι εντολές της υποενότητας «Άκυρη επιλογή λειτουργίας (γραμμές 285-288)».

(~) Έγκυρη επιλογή λειτουργίας (γραμμές 273-283)

Στις γραμμές «273-283» εκτελούνται οι εντολές της «if ch >= 1 && ch <= 4)» (γραμμή 273), για την περίπτωση εισαγωγής έγκυρης επιλογής λειτουργίας. Η διάσχιση της «if» έχει ως εξής :

Οι επιλογές λειτουργιών (γραμμές 275-283)

Στις γραμμές «275-283» υλοποιείται με μία εντολή ελέγχου «switch-case» το μενού με τις επιλογές λειτουργιών, όπου καλείται η συνήρτηση που εκτελεί την επιλεγμένη λειτουργία (το περιεχόμενο της ακέραιας μεταβλητής «ch» που διαβάζεται από τη «standard» είσοδο, κάθε φορά που εκτελείται ο βρόχος) και εφόσον, εκτελέστηκε χωρίς παραβιάσεις στους περιορισμούς, καταχωρείται σε μία μεταβλητή ένας χαρακτηριστικός αριθμός για τον υπολογισμό των έγκυρων επιλεγμένων λειτουργιών κάθε φορά που εκτελείται ο βρόχος. Η διάσχιση της «switch-case» έχει ως εξής :

[1] Κλήση της συνάρτησης "Check_Power (A, B)" (γραμμή 277)

Σε περίπτωση που η μεταβλητή «ch» περιέχει την τιμή «1» (λειτουργία [1] «A^B»), τότε, πραγματοποιείται η κλήση της συνάρτησης «Check_Power(…)», όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» (a_M) και «B» (b_M) και υλοποιείται στις γραμμές «80-98».

Καταμέτρηση της τιμής έγκυρης λειτουργίας [1] (γραμμή 278)

Επίσης, πραγματοποιείται η κλήση της συνάρτησης «Check_Valid_Power», όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» (a_M) και «B» (b_M) και υλοποιείται στις γραμμές «100-113», και υπολογίζεται με την παράσταση «m_P + Check_Valid_Power (a_M, b_M)» το πλήθος των έγκυρων λειτουργιών [1], κάθε φορά που εκτελείται ο βρόχος. Το αποτέλεσμα καταχωρείται στη μεταβλητή «m_P», όπου έχει για αρχική τιμή, την τιμή «0» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 260)». Η εντολή «break» μεταφέρει την ροή του προγράμματος στις επόμενες εντολές που βρίσκονται εκτός του σώματος της «switch-case».

[2] Κλήση της συνάρτησης "Check_Factorial (A, B)" (γραμμή 279)

Σε περίπτωση που η μεταβλητή «ch» περιέχει την τιμή «2» (λειτουργία [2] «A!» και «B!»), τότε, πραγματοποιείται η κλήση της συνάρτησης «Check_Factorial(…)», όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» (a_M) και «B» (b_M) και υλοποιείται στις γραμμές «168-175».

Καταμέτρηση της τιμής έγκυρης λειτουργίας [2] (γραμμή 280)

Επίσης, πραγματοποιείται η κλήση του αυτόνομου υποπρογράμματος «Check_Valid_Factorial(...)», όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» (a_M) και «B» (b_M) και υλοποιείται στις γραμμές «177-191», και υπολογίζεται με την παράσταση «m_F + Check_Valid_Factorial (a_M, b_M)» το πλήθος των έγκυρων λειτουργιών [2], κάθε φορά που εκτελείται ο βρόχος. Το αποτέλεσμα καταχωρείται στη μεταβλητή «m_F», όπου έχει για αρχική τιμή, την τιμή «0» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 261)». Η

εντολή «break» μεταφέρει την ροή του προγράμματος στις επόμενες εντολές που βρίσκονται εκτός του σώματος της «switch-case».

[3] Κλήση της συνάρτησης "Check_Combinations (A, B)" (γραμμή 281)

Σε περίπτωση που η μεταβλητή «ch» περιέχει την τιμή «3» (λειτουργία [3] «A! / B! * (A – B)!), τότε, πραγματοποιείται η κλήση της συνάρτησης «Check_Combinations(…)», όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» (a_M) και «B» (b_M) και υλοποιείται στις γραμμές «213-231».

Καταμέτρηση της τιμής έγκυρης λειτουργίας [3] (γραμμή 282)

Επίσης, πραγματοποιείται η κλήση του αυτόνομου υποπρογράμματος «Check_Valid_Combinations(...)», όπου δέχεται για είσοδο τους δύο ακέραιους αριθμούς «A» (a_M) και «B» (b_M) και υλοποιείται στις γραμμές «233-246», και υπολογίζεται με την παράσταση «m_C + Check_Valid_Combinations (a_M, b_M)» το πλήθος των έγκυρων λειτουργιών [3], κάθε φορά που εκτελείται ο βρόχος. Το αποτέλεσμα καταχωρείται στη μεταβλητή «m_C», όπου έχει για αρχική τιμή, την τιμή «0» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 262)». Η εντολή «break» μεταφέρει την ροή του προγράμματος στις επόμενες εντολές που βρίσκονται εκτός του σώματος της «switch-case».

(~) Άκυρη επιλογή λειτουργίας (γραμμές 285-288)

Στις γραμμές «285-288» εκτελούνται οι εντολές της «else» (γραμμή 285» που αντιστοιχεί στην «if ch >= 1 && ch <= 4)» (γραμμή 273), για την περίπτωση εισαγωγής άκυρης επιλογής λειτουργίας. Στη γραμμή «287» η εντολή «system (“cls”);» καθαρίζει τη «standard» έξοδο για την ομοίομορφη αποτύπωση του μενού.

[4] Κλήση της συνάρτησης "Exit ()" (γραμμή 294)

Σε περίπτωση που η μεταβλητή «ch» περιέχει την τιμή «4» (λειτουργία [4] Έξοδος), τότε, ο βρόχος σταματάει, γιατί, η παράσταση «ch != 4» παράγει αποτέλεσμα τη τιμή «False», πραγματοποιείται η κλήση της συνάρτησης «Exit(…)», όπου δεν δέχεται για είσοδο παραμέτρους και υλοποιείται στις γραμμές «248-253». Η τιμή που επιστρέφει, καταχωρείται στη μεταβλητή «m_E», όπου έχει για αρχική τιμή, την τιμή «0» (βλ. «Αρχικοποίηση μεταβλητών (γραμμή 263)».

Υπολογισμός του πλήθους των έγκυρων λειτουργιών (γραμμή 295)

Στη γραμμή «295» υπολογίζεται με την αριθμητική παράσταση «m_P + m_F + m_C + m_E» το πλήθος των έγκυρων λειτουργιών και το αποτέλεσμα καταχωρείται στη μεταβλητή «sum».

Εκτύπωση του πλήθους των έγκυρων λειτουργιών (γραμμή 296)

Στη γραμμή «296» τυπώνεται στη «standard» έξοδο το περιεχόμενο της μεταβλητής «sum» (το πλήθος των έγκυρων λειτουργιών) με μία συνάρτηση «printf()», συνοδευόμενο με το κατάλληλο μήνυμα.

ΠΑΡΑΔΕΙΓΜΑΤΑ

Παράδειγμα 1 (A == 9 / B == 10 / ch == 1, 2, 3, 4)

==========

Μενού αριθμητικών πράξεων με ακέραιους

==========

Εισάγετε τον ακέραιο αριθμό A : 9

Εισάγετε τον ακέραιο αριθμό B : 10

****

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 1

****

[1] Υπολογισμός της δύναμης A^B

A^B : 3486784512.000000

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 2

****

[2] Υπολογισμός του A! και του B!

A! : [362880]

B! : [3628800]

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 3

****

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

Σφάλμα

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 4

****

Αριθμός έγκυρων λειτουργιών : 3

Παράδειγμα 2 (A == 0 / B == 0 / ch == 1, 2, 3, 4)

==========

Μενού αριθμητικών πράξεων με ακέραιους

==========

Εισάγετε τον ακέραιο αριθμό A : 0

Εισάγετε τον ακέραιο αριθμό B : 0

****

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 1

****

[1] Υπολογισμός της δύναμης A^B

Σφάλμα

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 2

****

[2] Υπολογισμός του A! και του B!

A! : [1]

B! : [1]

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 3

****

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

Σφάλμα

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 4

****

Αριθμός έγκυρων λειτουργιών : 2

Παράδειγμα 3 (A == 5 / B == -2 / ch == 1, 2, 3, 1, 4)

==========

Μενού αριθμητικών πράξεων με ακέραιους

==========

Εισάγετε τον ακέραιο αριθμό A : 5

Εισάγετε τον ακέραιο αριθμό B : -2

****

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 1

****

[1] Υπολογισμός της δύναμης A^B

A^B : 0.040000

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 2

****

[2] Υπολογισμός του A! και του B!

A! : [120]

Σφάλμα

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 3

****

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

Σφάλμα

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 1

****

[1] Υπολογισμός της δύναμης A^B

A^B : 0.040000

---

[1] Υπολογισμός της δύναμης A^B

[2] Υπολογισμός του A! και του B!

[3] Υπολογισμός του πλήθους των συνδυασμών A ανά B

[4] Έξοδος

Επιλογή λειτουργίας : 4

****

Αριθμός έγκυρων λειτουργιών : 3

Παράδειγμα 4 (A == -7 / B == -6 / ch == 2, 4)

==========

Μενού αριθμητικών πράξεων με ακέραιους

==========

Εισάγετε τον ακέραιο αριθμό Α : -7

Εισάγετε τον ακέραιο αριθμό Β : -6

****

[1] Υπολογισμός της δύναμης Α^Β

[2] Υπολογισμός του Α! και του Β!

[3] Υπολογισμός του πλήθους των συνδυασμών Α ανά Β

[4] Έξοδος

Επιλογή λειτουργίας : 2

****

[2] Υπολογισμός του Α! και του Β!

Σφάλμα

Σφάλμα

---

[1] Υπολογισμός της δύναμης Α^Β

[2] Υπολογισμός του Α! και του Β!

[3] Υπολογισμός του πλήθους των συνδυασμών Α ανά Β

[4] Έξοδος

Επιλογή λειτουργίας : 4

****

Αριθμός έγκυρων λειτουργιών : 1

ΠΑΡΑΤΗΡΗΣΕΙΣ

Η ορθότητα των αποτελεσμάτων στα παραπάνω παραδείγματα ελέγχθηκε και με την χρήση τυπικής αριθμομηχανής και δεν διαπίστωθηκε καμία διαφορά.

Η συμβολοσειρά «**********» ενώνει τα περιεχόμενα που έχουν τυπωθεί της «standard» εξόδου, καθώς, η εντολή «system ("cls");» καθαρίζει τη «standard» έξοδο κάθε φορά που εκτελείται. Οι λειτουργίες με κόκκινο χρώμα είναι οι άκυρες και δεν υπολογίζονται στο πλήθος των έγκυρων λειτουργιών.

Αναλυτικά, έχουμε τρία παραδείγματα όπου το καθένα αντιστοιχεί σε ειδικές συνθήκες.

Παράδειγμα 1 (A == 9 / B == 10 / ch == 1, 2, 3, 4)

Στο «Παράδειγμα 1», ο χρήστης εισάγει πρώτο (A) τον ακέραιο «9», δεύτερο (B) τον ακέραιο «10» και επίλεγει τις εξής λειτουργίες :

[1] : A^B ➔ 9^10 ➔ 3486784512.000000

[2] : A! ➔ 9! ➔ 362880
B! ➔ 10! ➔ 3628800

[3] : A! / B! (A – B)! ➔ 9! / 10! * (9 – 10)! ➔ 9! / 10! * (-1)! ➔ Σφάλμα
Δεν προσδιορίζεται παραγοντικό με αρνητικό αριθμό

[4] : Έξοδος

Πλήθος έγκυρων λειτουργιών : [1] + [2] + [4] == 3

Παράδειγμα 2 (A == 0 / B == 0 / ch == 1, 2, 3, 4)

Στο «Παράδειγμα 2», ο χρήστης εισάγει πρώτο (A) τον ακέραιο «0», δεύτερο (B) τον ακέραιο «0» και επίλεγει τις εξής λειτουργίες :

[1] : A^B ➔ 0^0 ➔ Σφάλμα
Δεν προσδιορίζεται η δύναμη «0^0»

[2] : A! ➔ 0! ➔ 1
B! ➔ 0! ➔ 1

[3] : A! / B! (A – B)! ➔ 0! / 0! * (0 – 0)! ➔ 0! / 0! * 0! ➔ 1 / 1 * 1 ➔ Σφάλμα
Δεν προσδιορίζεται συνδυασμός με τον ίδιο ακέραιο π.χ 0 με 0

[4] : Έξοδος

Πλήθος έγκυρων λειτουργιών : [2] + [4] == 2

Παράδειγμα 3 (A == 5 / B == -2 / ch == 1, 2, 3, 1, 4)

Στο «Παράδειγμα 3», ο χρήστης εισάγει πρώτο (A) τον ακέραιο «5», δεύτερο (B) τον ακέραιο «-2» και επίλεγει τις εξής λειτουργίες :

[1] : A^B ➔ 5^(-2) ➔ 0.040000

[2] : A! ➔ 5! ➔ 120
B! ➔ (-2)! ➔ Σφάλμα
Δεν προσδιορίζεται παραγοντικό με αρνητικό αριθμό

[3] : A! / B! (A – B)! ➔ 5! / (-2)! * (5 – (-2)! ➔ Σφάλμα
Δεν προσδιορίζεται παραγοντικό με αρνητικό αριθμό

[1] : A^B ➔ 5^(-2) ➔ 3486784512.000000

[4] : Έξοδος

Πλήθος έγκυρων λειτουργιών : [1] + [2] + [3] + [4] == 4

Παράδειγμα 4 (A == -7 / B == -6 / ch == 2, 4)

Στο «Παράδειγμα 4», ο χρήστης εισάγει πρώτο (A) τον ακέραιο «-7», δεύτερο (B) τον ακέραιο «-6» και επίλεγει τις εξής λειτουργίες :

[2] : A! ➔ (-7)! ➔ Σφάλμα
Δεν προσδιορίζεται παραγοντικό με αρνητικό αριθμό
B! ➔ (-6)! ➔ Σφάλμα
Δεν προσδιορίζεται παραγοντικό με αρνητικό αριθμό

[4] : Έξοδος

Πλήθος έγκυρων λειτουργιών : [4] == 1

ΘΕΜΑ 3

ΕΠΙΣΗΜΑΝΣΗ

Το παρακάτω πρόγραμμα και η τεκμηρίωση του απαντούν στο ζητούμενο

του ερωτήματος «Θέμα 3».

ΠΡΟΓΡΑΜΜΑ

```
1 #include <stdio.h>
2
3 int MSum (int);
4
5 int main (int argc, char **argv)
6 {
7      system ("chcp 1253");
8
9      int x;
10     int p;
11
12     printf ("====================================\n\n");
13     printf ("Αναδρομική Συνάρτηση\n\n");
14     printf ("====================================\n\n");
15     printf ("Εισάγετε ακέραιο αριθμό : ");
16     scanf ("%d", &x);
17     p = MSum (x);
18     printf ("------------------------------------\n\n");
19     printf ("Ο ακέραιος αριθμός      : [%20d]\n", x);
20     printf ("Αποτέλεσμα συνάρτησης   : [%20d]\n\n", p);
21
22     return 0;
23 }
24
25 int MSum (int N)
26 {
27    if (N == 1)
28        return 1;
29    return N + MSum(N - 1);
30 }
```

ΤΕΚΜΗΡΙΩΣΗ

Το παραπάνω πρόγραμμα διαβάζει στην κύρια συνάρτηση "main()", με τη συνάρτηση "scanf()", από τη "standard" είσοδο έναν ακέραιο αριθμό "x" (γραμμή 16) και κάνει κλήση τη συνάρτηση "MSum()" με παράμετρο αυτόν τον αριθμό (γραμμή 17). Η συνάρτηση "MSum()" είναι τύπου "int", δηλαδή, επιστρέφει μία ακέραια τιμή στη γραμμή που έγινε η κλήση της και έχει για παράμετρο τον ακέραιο αριθμό "x" που διαβάστηκε από τη "standard" είσοδο. Ο αριθμός "x" έχει μεταβιβαστεί στη μεταβλητή "N" που αναγνωρίζει η συνάρτηση "MSum()". Η "MSum" περιέχει μία εντολή ελέγχου "if" (γραμμές 27-28), όπου ελέγχει αν η τιμή που παράγει η παράσταση "N == 1" παράγει αποτέλεσμα μία τιμή "True" (διάφορη του "0"). Εάν, ο αριθμός που διαβάστηκε είναι ο "1" (N == 1), τότε, η "MSum()" θα επιστρέψει με την εντολή "return 1" την τιμή "1". Ειδαλλώς, (N != 1) η ροή του προγράμματος μεταφέρεται στην αμέσως επόμενη εντολή που είναι η "return N + MSum(N - 1)". Η "MSum()" θα επιστρέψει με την εντολή αυτή, την τιμή της παράστασης "N + MSum(N - 1)".

Στη γραμμή "29" του προγράμματος η συνάρτηση "MSum()" κάνει κλήση τον εαυτό της και γι' αυτό τον λόγο θεωρείται αναδρομική συνάρτηση. Για παράδειγμα, έστω, ο αριθμός "5" που διαβάστηκε από τη "standard" είσοδο, στην κύρια συνάρτηση "main()". Η "main()" καλεί στη γραμμή "17" τη συνάρτηση "MSum()" με παράμετρο τον αριθμό "5". Η "MSum()" ελέγχει στη γραμμή αν η παράμετρος περιέχει την τιμή "1" (γραμμή 27), διακρίνει ότι δεν την περιέχει και έτσι καλεί τον εαυτό της με παράμετρο τον αριθμό "4" (N - 1).

Η παράσταση "N + MSum(N - 1)" με "N == 5" παράγει σαν αποτέλεσμα την τιμή "5 + 4 = 9". Η συνάρτηση καλείται αναδρομικά με παραμέτρους τους αριθμούς "3" (9 + 3 = 12), "2" (12 + 2 = 14) και "1". Με παράμετρο "1" ελέγχει αν "N == 1" (γραμμή 27) και επιστρέφει με την εντολή "return 1" την τιμή "1" που προστίθεται στο συνολικό αποτέλεσμα που επιστρέφει η "MSum()" (14 + 1 = 15). Επομένως, η "MSum()" με παράμετρο τον αριθμό "5" επιστρέφει την τιμή "15", δηλαδή, το άθροισμα της πρόσθεσης "1 + 2 + 3 + 4 + 5". Συνοψίζοντας, η "MSum()" για "N" ακέραια παράμετρο επιστρέφει το άθροισμα της πρόσθεσης "N + (N - 1) + (N - 2)+...+ 1".

ΘΕΜΑ 4

ΕΠΙΣΗΜΑΝΣΗ «Hanoi.c»

Το «Πρόγραμμα “Hanoi.c”» (Πηγαίος Κώδικας) και η «Τεκμηρίωση “Hanoi.c” (Ζητούμενο, Δομή, Συναρτήσεις, Μεταβλητές, Διάσχιση, Παραδείγματα) απαντούν στο ζητούμενο του ερωτήματος «Θέμα 2».

ΠΡΟΓΡΑΜΜΑ «Hanoi.c»

```
#include <stdio.h>

/* Δήλωση συναρτήσεων */
void Title (); // Ο τίτλος του προγράμματος
int Read_num_Disks (); // Εισαγωγή του πλήθους των δίσκων
void Print_num_Disks (int); // Εκτύπωση του αριθμού των δίσκων
int Num_Min_Moves (int); // Υπολογισμός των ελάχιστων κινήσεων του παιχνιδιού "Πύργοι του Ανόι"
void Print_Num_Min_Moves (int); // Εκτύπωση των ελάχιστων κινήσεων του παιχνιδιού "Πύργοι του Ανόι"
void Move_Disks (int, char, char, char);  // Η διαδρομή των δίσκων

/* Όπου N ο αριθμός δίσκων, Α ο στύλος 1, Β ο στύλος 2 και Γ ο στύλος 3 */

int main (int argc, char **argv)  /* main (int argc, char **argv) */
{
	system ("chcp 1253");

	int n; // Δήλωση μεταβλητών
	char pole_1, pole_2, pole_3;

	pole_1 = 'A'; // Στύλος 1
	pole_2 = 'B'; // Στύλος 2
	pole_3 = 'Γ'; // Στύλος 3

	Title (); // Κλήση της συνάρτησης "Title()"
	n = Read_num_Disks (); // Κλήση της συνάρτησης "Read_num_Disks()"
	Print_num_Disks (n); // Κλήση της συνάρτησης "Print_num_Disks (N)"
	Print_Num_Min_Moves (n); // Κλήση της συνάρτησης "Print_Num_Min_Moves (N)"
	Move_Disks (n, pole_1, pole_2, pole_3);  // Κλήση της συνάρτησης "Move_Disks (N, A, B, Γ)"

	return 0;
}

void Title () /* Title () */
```

```
{
	printf ("============================================================
\n\n");
	printf ("Πύργοι του Ανόι\n\n"); // Τίτλος του προγράμματος
	printf ("============================================================
\n\n");
}

int Read_num_Disks () /* Read_num_Disks () */
{
	int n_RnD; // Δήλωση μεταβλητών

	printf ("Εισάγετε αριθμό δίσκων : ");
	scanf ("%d", &n_RnD); // Εισαγωγή του πλήθους των δισκών
	printf ("\n------------------------------------------------------------
\n\n");

	return n_RnD; // Επιστροφή του πλήθους των δισκών
}

void Print_num_Disks (int n_PnD) /* Print_num_Disks (N) */
{
	printf ("Πλήθος δισκών      : [%20d]\n", n_PnD);  // Εκτύπωση του
πλήθους των δισκών
}

int Num_Min_Moves (int n_NMM) /* Num_Min_Moves (N) */
{
	int min_moves; // Δήλωση μεταβλητών

	if (n_NMM == 0) /* (~) 0 δίσκοι */
	{
		return 0; // Επιστροφή πλήθους ελάχιστων κινήσεων
	}
	else  /* (~) 0< δίσκοι */
	{
		min_moves = Num_Min_Moves (n_NMM - 1) + 1 + Num_Min_Moves (n_NMM
- 1);   // Υπολογισμός πλήθους ελάχιστων κινήσεων
```

```
		return min_moves; // Επιστροφή πλήθους ελάχιστων κινήσεων
	}
}

void Print_Num_Min_Moves (int n_PNMM) /* Print_Num_Min_Moves */
{
	int print_min_moves; // Δήλωση μεταβλητών

	print_min_moves = Num_Min_Moves (n_PNMM); // Κλήση της συνάρτησης
"Num_Min_Moves (N)"
	printf ("Ελάχιστες κινήσεις : [%20d]\n\n", print_min_moves);
// Εκτύπωση του πλήθους ελάχιστων κινήσεων
}

void Move_Disks (int n_MD, char pole_A, char pole_B, char pole_C)
/* Move_Disks (N, A, B, Γ) */
{
	if (n_MD == 1 && n_MD != 0) /* (~) 1 δίσκος */
	{
		printf ("%c --> %c\n", pole_A, pole_C); // Διαδρομή
	}
	else  /* (~) 1< δίσκοι */
	{
		if (n_MD != 1 && n_MD != 0) /* (+) 1< δίσκοι */
		{
			n_MD = n_MD - 1; // Αφαίρεση του πλήθους των δισκών
			Move_Disks (n_MD, pole_A, pole_C, pole_B);
// Αναδρομική κλήση της συνάρτησης "Move_Disks (N-1, A, Γ, B)"
			printf ("%c --> %c\n", pole_A, pole_C);
// Διαδρομή
			Move_Disks (n_MD, pole_B, pole_A, pole_C);
// Αναδρομική κλήση της συνάρτησης "Move_Disks (N-1, B, A, Γ)"
		}
	}

}
```

ΤΕΚΜΗΡΙΩΣΗ «Hanoi.c»

ΖΗΤΟΥΜΕΝΟ

Το πρόγραμμα «Hanoi.c» επιτυγχάνει τις εξής λειτουργίες:

a) Διαβάζει από τη «standard» το πλήθος των δίσκων.

b) Τυπώνει στη «standard» έξοδο το πλήθος των δίσκων.

c) Υπολογίζει το πλήθος των ελάχιστων κινήσεων από τον στύλο «Α» στον στύλο «Γ».

d) Τυπώνει στη «standard» έξοδο το πλήθος των ελάχιστων κινήσεων από τον στύλο «Α» στον στύλο «Γ».

e) Εκτελεί τον αλγόριθμο «Πύργοι του Ανόι» με την βοήθεια της αναδρομική συνάρτησης.

f) Τυπώνει στη «standard» έξοδο τον αλγόριθμο «Πύργοι του Ανόι» με την βοήθεια της αναδρομική συνάρτησης.

ΔΟΜΗ

Προκειμένου να υλοποιηθεί το ζητούμενο χρησιμοποίηθηκαν, αρχικά, οι βιβλιοθήκες (.h) :

a) «stdio.h»: Περιέχει τις έτοιμες συναρτήσεις «scanf(...)» και «printf(...)» που συνδέονται με τα κανάλια εισόδου και εξόδου αντίστοιχα για την ανάγνωση και την τύπωση περιεχομένων των αντίστοιχων μεταβλητών. Επίσης, η «printf(...)» χρησιμοποιήθηκε για να τυπωθούν χαρακτηριστικά μηνύματα για την βέλτιστη κατανόηση του πηγαίου κώδικα.

Επιπρόσθετα, χρησιμοποιήθηκαν οι χαρακτηριστικοί τελεστές :

a) αριθμητικοί : +, -

b) σχεσιακοί : ==, !=,

c) λογικοί : &&

d) ανάθεσης : =

e) τελεστής & : Για την διεύθυνση μεταβλητής ως δεύτερο όρισμα της

συνάρτησης «scanf()» που συνδέεται με τη «standard» είσοδο

Οι εντολές ελέγχου :

a) if – else

Η κάθε λειτουργία απ' την ενότητα «Ζητούμενο» υλοποιήθηκε με αυτόνομα υποπρογράμματα (βλ. ενότητα «Συναρτήσεις»).

ΣΥΝΑΡΤΗΣΕΙΣ

Βλ. σχόλια γραμμές «2-9»

ΜΕΤΑΒΛΗΤΕΣ

Βλ.

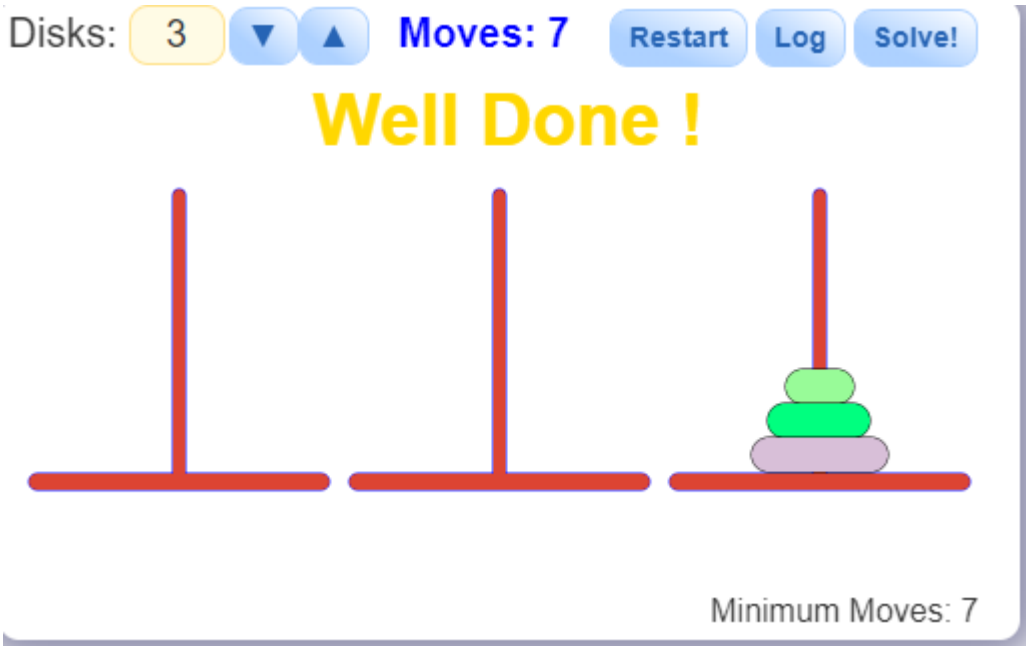
γραμμές «15-16» (main)
γραμμή «40» (Read_num_Disks)
γραμμή «56» (Num_Min_Moves)
γραμμή «72» (Print_Num_Min_Moves)

ΔΙΑΣΧΙΣΗ

Βλ. Σχόλια «ΠΡΟΓΡΑΜΜΑ "Hanoi.c"»

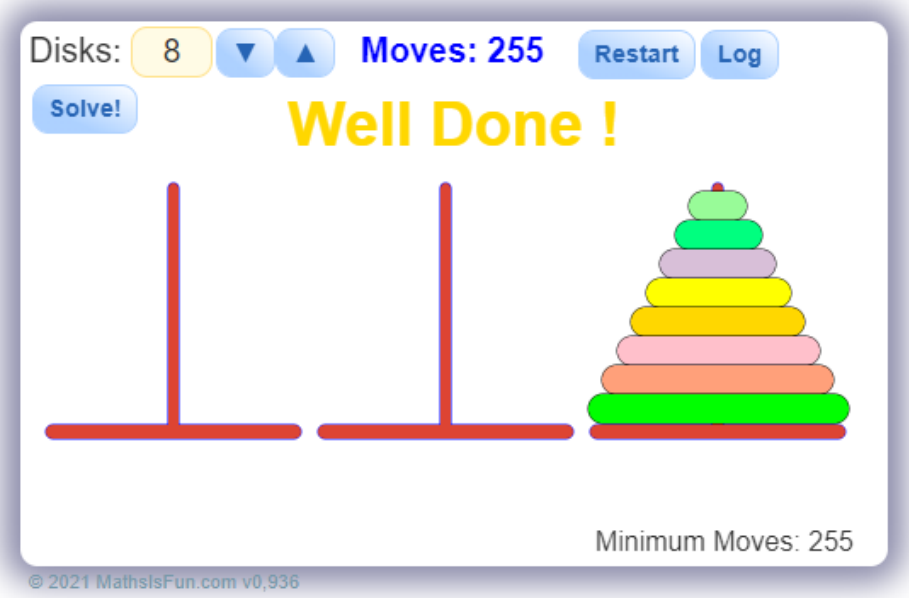
ΠΑΡΑΔΕΙΓΜΑΤΑ / ΠΑΡΑΤΗΡΗΣΕΙΣ

```
============================================================

Πύργοι του Ανόι

============================================================

Εισάγετε αριθμό δίσκων : 3

------------------------------------------------------------

Πλήθος δισκών      : [                    3]
Ελάχιστες κινήσεις : [                    7]

A --> Γ
A --> B
Γ --> B
A --> Γ
B --> A
B --> Γ
A --> Γ
```



Tower of Hanoi

Object of the game is to move all the disks over to Tower 3 (with your mouse).
But you cannot place a larger disk onto a smaller disk.



Σας ευχαριστώ για την προσοχή σας.