Η Βιβλιοθήκη i2p

1. Βιβλιοθήκη τρίτου κατασκευαστή (Third-Party library)

Third-Party library είναι μια βιβλιοθήκη που έχει αναπτυχθεί από άλλον προγραμματιστή. Η σημασία αξιοποίησης βιβλιοθήκης τρίτου κατασκευαστή είναι εμφανής από το παρακάτω απόσπασμα.

"In computer programming, a **third-party software component** is a reusable software component developed to be either freely distributed or sold by an entity other than the original vendor of the development platform. The third-party software component market thrives because many programmers believe that **component-oriented development** improves the efficiency and the quality of developing custom applications." Wikipedia

2. Αντικείμενο και Στόχος της βιβλιοθήκη i2p

Η βιβλιοθήκη i2p αναπτύχθηκε για δύο λόγους:

1. για να σας δώσει τη δυνατότητα να εξοικειωθείτε με τη διαδικασία αξιοποίησης βιβλιοθήκης τρίτου κατασκευαστή, και,
2. να σας διευκολύνει στην ανάπτυξη των πρώτων εκδόσεων ορισμένων προγραμμάτων.

3. Αξιοποίηση βιβλιοθήκης

Για να αξιοποιήσετε μια βιβλιοθήκη θα πρέπει να κάνετε τις ενέργειες Ε1-Ε4 όπως περιγράφονται παρακάτω:

3.1 Ε1-Κατεβάστε την βιβλιοθήκη

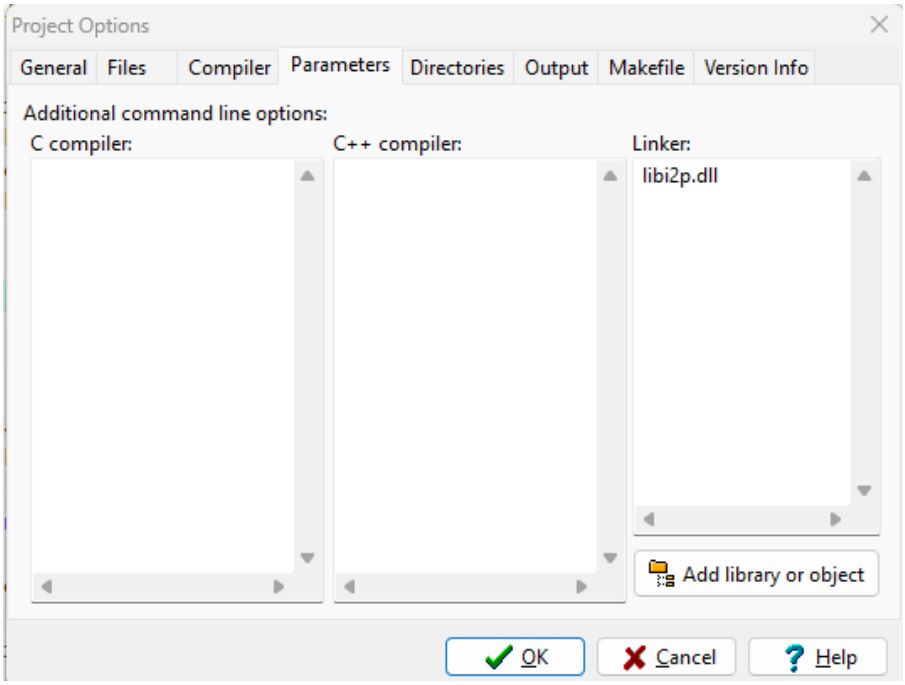
Κατεβάστε και τοποθετήστε την βιβλιοθήκη (αρχείο .h και αρχείο .dll) στο ευρετήριο του project σας.

3.2 Ε2-Συμπεριλάβετε το αρχείο επικεφαλίδας (#include)

Συμπεριλάβετε με την εντολή προεπεξεργαστή #include στο αρχείο πηγαίου κώδικα σας το αρχείο επικεφαλίδας, π.χ. #include "i2p.h".
Τοποθετούμε το header file σε " " για να ψάξει ο προεπεξεργαστής (pre-processor) το αρχείο στο κατάλογο (ευρετήριο) του προγράμματος μας.
Σε < > περικλείονται τα αρχεία επικεφαλίδας της βασικής βιβλιοθήκης για να δηλώσουν στον επεξεργαστή πως αυτά θα τα αναζητήσει στον κατάλογο του compiler που χρησιμοποιείτε.

3.3 Ε3-Ενημερώστε το IDE σας για το αρχείο .dll

Για παράδειγμα για να ενημερώσετε το DevC++ επιλέξτε
Project->Projects Options>Parameters και
στην συνέχεια `Add library or Object` (Σχήμα 1). Επιλέξτε από το ευρετήριο του project το αρχείο της βιβλιοθήκης. Θα το δείτε στο πλαίσιο κάτω από τον Linker καθώς το αρχείο αυτό θα το αξιοποιήσει, μετά την μεταγλώττιση, ο Linker. Επιλέξτε OK.



Σχήμα 1. Ενημέρωση IDE για το αρχείο lib12p.dll

3.4 E4-Καλέστε συνάρτηση της βιβλιοθήκης

Καλέστε μια συνάρτηση της βιβλιοθήκης πάντα σύμφωνα με το function prototype της που θα βρείτε στο αρχείο επικεφαλίδας της βιβλιοθήκης. Στην επόμενη ενότητα δίνονται οι συναρτήσεις που περιέχει η βιβλιοθήκη και στην μεθεπόμενη οδηγίες και παραδείγματα κλήσης των συναρτήσεων.

4. Οι συναρτήσεις της βιβλιοθήκης i2p

4.1 Συναρτήσεις Εισόδου

4.1.1 Περιγραφή συναρτήσεων

Στην ομάδα αυτή των συναρτήσεων περιλαμβάνονται συναρτήσεις για είσοδο δεδομένων από την βασική είσοδο. Τα function prototypes των συναρτήσεων εισόδου, τα οποία βρίσκονται στο αρχείο επικεφαλίδας (header file) i2p.h, είναι τα παρακάτω:

```
int getInt(char message[]);
int getArrayOfInts(int ar[],int maxItems);
float getFloat(char message[]);

char getChar(char message[]);
void getStringV1(char message[], char str[]);
void getStringV2(char message[],char str[], int strSize);
```

Η συνάρτηση **getArrayOfInts** εμφανίζει το μήνυμα message στην βασική έξοδο (stdout) και διαβάζει από την stdin maxItems ακεραίους τους οποίους τοποθετεί στον πίνακα ar.

Η συνάρτηση **getStringV2** εμφανίζει το μήνυμα message στην βασική έξοδο (stdout) και διαβάζει από την stdin ένα αλφαριθμητικό με μέγιστο αριθμό χαρακτήρων strSize και το βάζει στον πίνακα χαρακτήρων str.

Η **getStringV1** διαβάζει χαρακτήρες χωρίς έλεγχο και είναι ευθύνη του προγραμματιστή να έχει δεσμεύσει αρκετή μνήμη.

4.1.2 Δράση 1

Αναπτύξτε ένα πρόγραμμα που θα επιδεικνύει την χρήση των συναρτήσεων Εισόδου της βιβλιοθήκης i2p.

4.2 Συναρτήσεις ταξινόμησης

4.2.1 Περιγραφή συναρτήσεων

Στην ομάδα αυτή των συναρτήσεων περιλαμβάνονται συναρτήσεις για ταξινόμηση πίνακα αλφαριθμητικών και πίνακα ακεραίων.

Τα function prototypes των συναρτήσεων ταξινόμησης πίνακα αλφαριθμητικών, τα οποία βρίσκονται στο αρχείο επικεφαλίδας (header file) i2p.h είναι τα παρακάτω:

```
void sortStringArrayInc(char base[],int numOfElements, int strWidth);
void sortStringArrayDec(char base[],int numOfElements, int strWidth);

void sortIntArrayInc(int ar[],int numOfElements);
void sortIntArrayDec(int ar[],int numOfElements);
```

Η συνάρτηση `sortStringArrayInc` ταξινομεί ένα πίνακα αλφαριθμητικών με αύξουσα σειρά ενώ η `sortStringArrayDec` με φθίνουσα.

Και οι δύο συναρτήσεις δέχονται ως ορίσματα τα:

`base`: δείκτης στο πρώτο αλφαριθμητικό του πίνακα.

`numOfElements`: ο αριθμός των στοιχείων (αλφαριθμητικών) του πίνακα.

`strWidth`:Το μέγεθος σε bytes του στοιχείου του πίνακα, δηλαδή του αλφαριθμητικού.

Τα function prototypes των συναρτήσεων ταξινόμησης πίνακα ακεραίων, τα οποία βρίσκονται στο αρχείο επικεφαλίδας (header file) i2p.h είναι τα παρακάτω:

```
void sortIntArrayInc(int ar[],int numOfElements);
void sortIntArrayDec(int ar[],int numOfElements);
```

Η συνάρτηση `sortIntArrayInc` ταξινομεί ένα πίνακα ακεραίων με αύξουσα σειρά ενώ η `sortIntArrayDec` με φθίνουσα.

Και οι δύο συναρτήσεις δέχονται ως ορίσματα τα:

`ar` : πίνακας ακεραίων (δείκτης στο πρώτο στοιχείο (ακέραιο) του πίνακα).

`numOfElements` : ο αριθμός των στοιχείων (ακεραίων) του πίνακα.

Το αρχείο επικεφαλίδας i2p.h της βιβλιοθήκης i2p δίνεται στο Σχήμα 2.

```
1   #include <string.h>
2   #include <search.h>
3   #include <stdbool.h>
4   // i2p library Version 6.1
5   // Author: Kleanthis Thramboulidis
6   typedef struct fraction{
10
11  typedef struct expression{
16
17  int getInt(char message[]);
18  int getArrayOfInts(int ar[],int maxItems);
19  float getFloat(char message[]);
20
21  char getChar(char message[]);
22  void getStringV1(char message[], char str[]);
23  void getStringV2(char message[],char str[], int strSize);
24
25  void sortStringArrayInc(char base[],int numOfElements, int strWidth);
26  void sortStringArrayDec(char base[],int numOfElements, int strWidth);
27
28  void sortIntArrayInc(int ar[],int numOfElements);
29  void sortIntArrayDec(int ar[],int numOfElements);
```

Σχήμα 2. Το αρχείο επικεφαλίδας i2p.h της 3ης έκδοσης της βιβλιοθήκης i2p.

4.2.2 Δράση 2

Αναπτύξτε ένα πρόγραμμα που θα επιδεικνύει την χρήση των συναρτήσεων ταξινόμησης της βιβλιοθήκης i2p. Ταξινομήστε σε αύξουσα και φθίνουσα σειρά ένα πίνακα ακεραίων και ένα πίνακα αλφαριθμητικών.

4.3 Η συνάρτηση getExpression

Η συνάρτηση δίνεται σε δύο εκδόσεις: `getExpressionV1()` και `getExpressionV2()`.
Η συνάρτηση και στις 2 εκδόσεις της :

a) ζητάει από τον χρήση να εισάγει από την βασική είσοδο μια έκφραση όπως αυτή ορίζεται παρακάτω, και,
b) διαβάζει από την βασική είσοδο τα στοιχεία της έκφρασης που εισάγει ο χρήστης.

Στα στοιχεία αυτά θα πρέπει με κάποιο τρόπο να αποκτήσει πρόσβαση η συνάρτηση που κάλεσε την readExpression. Στο σημείο αυτό είναι η διαφορά των 2 εκδόσεων της `getExpression`.

Η **έκφραση είναι της μορφής**

<τελεστής> <αριθμητής 1ου κλάσματος>/<παρονομαστής 1ου κλάσματος> <αριθμητής 2ου κλάσματος>/<παρονομαστής 2ου κλάσματος>

Όπου ο τελεστής μπορεί να είναι + - * /
Παραδείγματα εκφράσεων
+ 1/5 3/4
* 3/6 4/12

4.3.1 Η συνάρτηση getExpression με δείκτες

Η `getExpressionV1()` διαβάζει τα στοιχεία που αποτελούν μια έκφραση προθεματικού τελεστή (prefix notation παράγραφος 4.2.1) με κλάσματα, δηλαδή τον τελεστή και τους αριθμητές και παρονομαστές των κλασμάτων. Παράδειγμα έκφρασης: + 2/13 5/13

Η συνάρτηση δέχεται ως ορίσματα δείκτες σε μεταβλητές όπου θα βάλει τις τιμές που θα διαβάσει από την κύρια είσοδο. Επιστρέφει δε:

- false αν στην θέση του τελεστή δοθεί ο χαρακτήρας q η Q,
- true σε κάθε άλλη περίπτωση

Η συνάρτηση `getExpressionV1`έχει το παρακάτω function prototype

```
bool getExpressionV1(char *operatorPtr, int *op1nPtr,int
*op1dPtr,int *op2nPtr,int *op2dPtr);
```

όπου
`operatorPtr` : operator Pointer (Δείκτης στην μεταβλητή operator)
`op1nPtr` : operand1 numerator Pointer (Δείκτης στην μεταβλητή operand1 numerator)
`op1dPtr` : operand1 denominator Pointer (Δείκτης στην μεταβλητή operand1 denominator)
`op2nPtr` : operand2 numerator Pointer
`op2dPtr` : operand2 denominator Pointer

Η συνάρτηση διαβάζει τα στοιχεία που αποτελούν μια έκφραση προθεματικού τελεστή (prefix notation) (παράγραφος 4.2.1) με κλάσματα, όπως για παράδειγμα η + 2/13 5/13. Διαβάζει δηλαδή τον τελεστή και τους αριθμητές και παρονομαστές των κλασμάτων. Επειδή δεν μπορεί με τον μηχανισμό της επιστρεφόμενης τιμής να επιστρέψει όλα αυτά στην συνάρτηση που την κάλεσε, η συνάρτηση δέχεται ως ορίσματα δείκτες σε μεταβλητές όπου θα βάλει τις τιμές που θα διαβάσει από την κύρια είσοδο. Τον περιορισμό αυτό που βάζει ο μηχανισμός της επιστρεφόμενης τιμής θα τον παρακάμψουμε στην επόμενη έκδοση της συνάρτησης η οποία θα αξιοποιεί τις δομές (struct).

4.3.2 Η συνάρτηση `getExpression` με δομές

Το function prototype της έκδοσης αυτής είναι το παρακάτω:

```
Expression getExpressionV2(void);
```

Παρατηρούμε πως η έκδοση αυτή της συνάρτησης δεν δέχεται κανένα όρισμα αλλά αξιοποιεί την κατασκευή της δομής (struct) για να επιστρέψει στην συνάρτηση που την κάλεσε τα στοιχεία της έκφρασης τα οποία διαβάζει από την κύρια είσοδο (stdin).
Αν στην θέση του τελεστή δοθεί από τον χρήστη ο χαρακτήρας q η Q, η συνάρτηση επιστρέφει έχοντας τοποθετήσει τον χαρακτήρα στην θέση του τελεστή (exp.oprtr). Αυτό μπορείτε να το ελέγξετε για τερματισμό του προγράμματος.

Το αρχείο επικεφαλίδας i2p.h εκτός από το πρωτότυπο της συνάρτησης περιέχει και τον ορισμό της δομής `Expression` ο οποίος δίνεται στο Σχήμα 3.

```
1   #include <string.h>
2   #include <search.h>
3   // i2p library Version 6.0
4   // Author: Kleanthis Thramboulidis
5   typedef struct fraction{
6       int ar;
7       int par;
8   }Fraction;
9
10  typedef struct expression{
11      char operator;
12      Fraction op1;
13      Fraction op2;
14  }Expression;

31  bool getExpressionV1(char *operatorPtr,int *op1np,int *op1dp,int *op2np,int *op2dp); // r
32  Expression getExpressionV2(void); // reads an expression of the following form + 1/5 3/15
```

Σχήμα 3. Το αρχείο επικεφαλίδας i2p.h της βιβλιοθήκης i2p.

4.3.3 Δράση 3

Αναπτύξτε ένα πρόγραμμα που θα επιδεικνύει την χρήση των δύο εκδόσεων της συνάρτησης `readExpression`(). Ένα παράδειγμα είναι η Εργαστηριακή Άσκηση Fractions. Το πρόγραμμα σας να αξιοποιεί και τις δομές `Fraction` `Expression` που ορίζει η i2p.

5. Παραδείγματα κλήσης συναρτήσεων

5.1. Συνάρτηση ταξινόμησης πίνακα ακεραίων

Αν υποθέσουμε πως έχουμε τον παρακάτω πίνακα ακεραίων `ar` ο οποίος περιέχει 10 ακεραίους (int).

```
int ar[10];
```

Στην περίπτωση αυτή η κλήση της `sortIntArrayInc` διαμορφώνεται όπως παρακάτω:

```
sortIntArrayInc(ar, sizeof(ar)/sizeof(ar[0]));
```

Τα πραγματικά ορίσματα περιγράφονται στη συνέχεια. Το `ar` είναι ο δείκτης στο πρώτο στοιχείο του πίνακα. Το `sizeof(ar)/sizeof(ar[0])` μας δίνει το πλήθος των στοιχείων του πίνακα και το `sizeof(ar[0])` μας δίνει το μέγεθος σε bytes του στοιχείου του πίνακα, δηλαδή στην περίπτωση μας του τύπου int.

5.2. Συνάρτηση ταξινόμησης πίνακα αλφαριθμητικών

Αν υποθέσουμε πως έχουμε τον παρακάτω πίνακα αλφαριθμητικών strAr ο οποίος περιέχει 100 αλφαριθμητικά μέγιστου πλάτους 40.

```
char strAr[100][40];
```

Στην περίπτωση αυτή η κλήση της `sortStringArrayInc` διαμορφώνεται όπως παρακάτω:

```
sortStringArrayInc ((char *)strAr, sizeof(strAr)/sizeof(strAr[0]),
                sizeof(strAr[0]));
```

Τα πραγματικά ορίσματα περιγράφονται στη συνέχεια. Το `strAr` είναι ο δείκτης στο πρώτο στοιχείο του πίνακα. Το `sizeof(strAr)/sizeof(strAr[0])` μας δίνει το πλήθος των στοιχείων του πίνακα και το `sizeof(strAr[0])` μας δίνει το μέγεθος σε bytes του στοιχείου του πίνακα.

5.3. getExperssionV1()

Η 1η έκδοση της `getExperssion` έχει το παρακάτω function prototype:

```
void getExpressionV1(char *operatorPtr, int *op1nPtr,int *op1dPtr,
                        int *op2nPtr,int *op2dPtr);
```

Το Σχήμα 4 δίνει τον πηγαίο κώδικα από ένα παράδειγμα αξιοποίησης της συνάρτησης `getExpressionV1()`. Προσέξτε τη δήλωση των μεταβλητών για τα στοιχεία της έκφρασης ως γενικές μεταβλητές για να τι βλέπει και η displayExpression. Το Σχήμα 5 δίνει ένα screenshot από μία εκτέλεση του προγράμματος.

```
73      int op1n,op1d,op2n,op2d;
74      char oprtr;
75      bool choise;
76      Expression exp;
77
78      while(getExpressionV1(&oprtr,&op1n,&op1d,&op2n,&op2d)) {
79          displayExpressionV1(oprtr,op1n,op1d,op2n,op2d);
80      }
```

```
90  void displayExpressionV1(char oprtr,int op1n,int op1d,int op2n,int
91      printf("\nExpression to evaluate: %c %d/%d  %d/%d\n",oprtr,op1
92  }
```

Σχήμα 4. Παράδειγμα κλήσης της `getExpressionV1()` της i2p.

```
Enter the expression (e.g. + 1/4 2/4) or q (for termination): *1/2 3/4

Expression to evaluate: * 1/2  3/4
Enter the expression (e.g. + 1/4 2/4) or q (for termination): + 5/6 7/8

Expression to evaluate: + 5/6  7/8
Enter the expression (e.g. + 1/4 2/4) or q (for termination): q

--------------------------------
Process exited after 15.34 seconds with return value 0
Press any key to continue . . .
```

Σχήμα 5. Screenshot εκτέλεσης του προγράμματος κλήσης της `getExpressionV1()` της i2p.

5.4. getExperssionV2()

Η 2 έκδοση της `getExpression` έχει το παρακάτω function prototype:

```
Expression getExpressionV2(void);
```

Το Σχήμα 6 δίνει τον πηγαίο κώδικα από ένα παράδειγμα αξιοποίησης της συνάρτησης `getExperssionV2()`. Προσέξτε τη δήλωση της μεταβλητής για τα στοιχεία της έκφρασης ως τοπικής μεταβλητής στην main οπότε δεν την βλέπει η displayExpression και αναγκαστικά την περνάμε ως όρισμα σε αυτήν κατά την κλήση της. Το Σχήμα 7 δίνει ένα screenshot από μία εκτέλεση του προγράμματος.

```
73      int op1n,op1d,op2n,op2d;
74      char oprtr;
75      bool choise;
76      Expression exp;
77
78  //  while(getExpressionV1(&oprtr,&op1n,&op1d,&op2n,&op2d)) {
79  //      displayExpressionV1(oprtr,op1n,op1d,op2n,op2d);
80  //  }
81      while (true){
82          exp=getExpressionV2();
83          if((exp.oprtr=='q')||(exp.oprtr=='Q'))
84              break;
85          displayExpressionV2(exp);
86      }
87      return 0;
```

Σχήμα 6. Παράδειγμα κλήσης της `getExpressionV2()` της i2p.

```
Enter the expression (e.g. + 1/4 2/4): * 1/2 34/124

Expression to evaluate: * 1/2  34/124
Enter the expression (e.g. + 1/4 2/4): +34/12 5/6

Expression to evaluate: + 34/12  5/6
Enter the expression (e.g. + 1/4 2/4): q

--------------------------------
Process exited after 26.83 seconds with return value 0
Press any key to continue . . .
```

Σχήμα 7. Screenshot εκτέλεσης του προγράμματος κλήσης της της `getExpressionV2()` της i2p.