

ΕΡΓΑΣΤΗΡΙΟ 3: Προγραμματιστικά Περιβάλλοντα και το Πρώτο Πρόγραμμα C

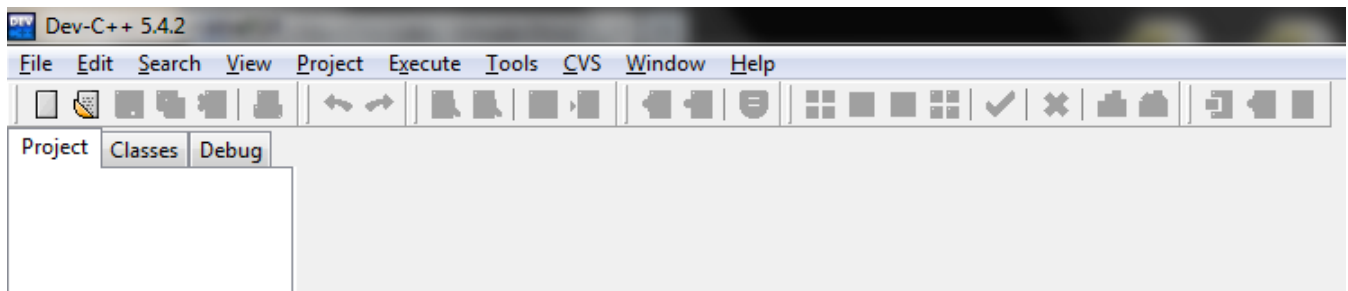
Στο εργαστήριο αυτό, θα ασχοληθούμε με δύο προγραμματιστικά περιβάλλοντα για τη γλώσσα C: τον gcc μεταγλωττιστή της C σε περιβάλλον Linux, και το περιβάλλον ανάπτυξης Dev-C++ για Windows. Επίσης, θα χρησιμοποιήσουμε και το πρόγραμμα WinSCP για να μεταφέρουμε αρχεία από υπολογιστές Windows στον λογαριασμό μας στην σχολή (και αντίστροφα). Τέλος, θα ασχοληθούμε με το πρώτο πρόγραμμα μας σε γλώσσα C, το οποίο θα μεταγλωττίσουμε, θα εκτελέσουμε και θα πειραματιστούμε με την έξοδό του.


1. Το περιβάλλον προγραμματισμού Dev-C++

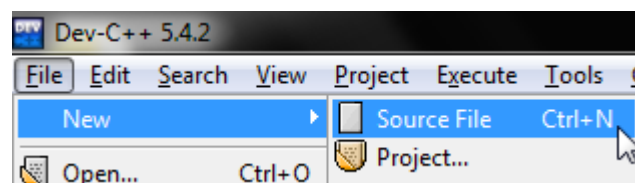
Στους λογαριασμούς των Windows συστημάτων της σχολής είναι διαθέσιμο το IDE Dev-C++, ένα γραφικό περιβάλλον ανάπτυξης κώδικα που απλοποιεί τη διαδικασία συγγραφής και μεταγλώττισης κώδικα (“αποκρύπτοντάς” μας την ύπαρξη και λειτουργία του gcc μεταγλωττιστή). Μπορούμε να το εγκαταστήσουμε στον προσωπικό μας υπολογιστή, κατεβάζοντας την τελευταία έκδοση από εδώ:

<http://sourceforge.net/projects/orwelldevcpp/files/latest/download>

Γράφουμε Dev-C++ και εκτελούμε στην έναρξη ή επιλέγουμε το σχετικό εικονίδιο από το μενού προγραμμάτων για να ανοίξουμε το περιβάλλον προγραμματισμού Dev-C++.



Για να δημιουργήσουμε ένα νέο αρχείο κώδικα, κάνουμε κλικ στο “File” -> “New” -> “Source File”, ή στο εικονίδιο .

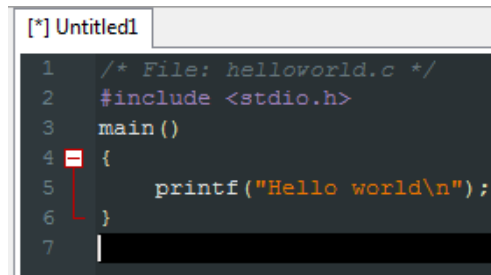


Στο κέντρο της οθόνης δημιουργείται ένα κενό αρχείο με όνομα καρτέλας “Untitled1” στο οποίο μπορούμε να πληκτρολογήσουμε τον κώδικα του πρώτου προγράμματός μας!


Ας γράψουμε εδώ λοιπόν, το πρώτο μας πρόγραμμα C:


```
/* File: helloworld.c */  
  
#include <stdio.h>  
  
main()  
{  
    printf("Hello world\n");  
}
```

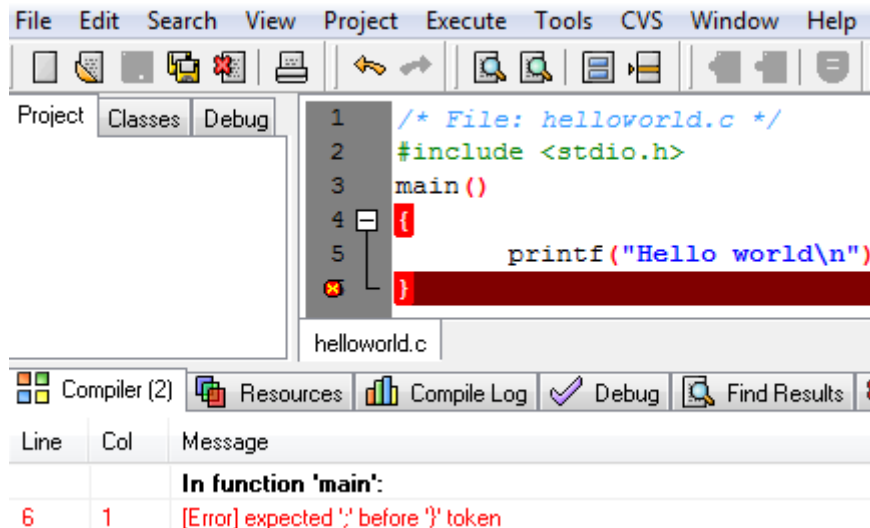
Αφού ολοκληρώσουμε την πληκτρολόγηση, πρέπει να αποθηκεύσουμε το αρχείο κώδικα στον δίσκο μας (το αστέρι στα αριστερά της καρτέλας υποδηλώνει πως έχουν γίνει αλλαγές που δεν έχουν σωθεί).



```
[*] Untitled1
1  /* File: helloworld.c */
2  #include <stdio.h>
3  main()
4  {
5      printf("Hello world\n");
6  }
7
```

Για να το αποθηκεύσουμε, επιλέγουμε “File” -> “Save” ή το κουμπί , ώστε να ανοίξει το σχετικό παράθυρο διαλόγου. Εκεί, πρώτα πηγαίνουμε στη λίστα “Save as type” και επιλέγουμε “C source files (*.c)”. Έπειτα, πλοηγούμαστε στη τοποθεσία (π.χ. Επιφάνεια Εργασίας) όπου μας ενδιαφέρει να αποθηκεύσουμε το αρχείο. Τέλος, δίνουμε ένα κατάλληλο όνομα στο αρχείο (π.χ. helloworld.c) και αποθηκεύουμε.

Για να μεταγλωττίσουμε το πρόγραμμά μας, επιλέγουμε “Execute” -> “Compile” ή πατάμε το F9, ή πατάμε το εικονίδιο . Αν το πρόγραμμά μας έχει συντακτικά λάθη ή επισημάνσεις, τότε στο κάτω μέρος της οθόνης θα εμφανίζονται σχετικά μηνύματα του μεταγλωττιστή, τα οποία περιγράφουν τη φύση του προβλήματος και μας καθοδηγούν για την επίλυσή του. Αφού εκτελέσουμε το μεταγλωττιστή, τότε θα εμφανιστούν μηνύματα όπως στην ακόλουθη εικόνα.

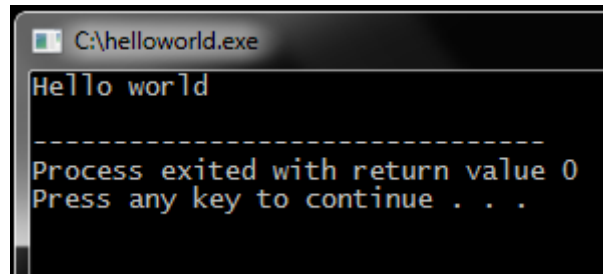


```
File Edit Search View Project Execute Tools CVS Window Help
Project Classes Debug
1  /* File: helloworld.c */
2  #include <stdio.h>
3  main()
4  {
5      printf("Hello world\n"
6
Compiler (2) Resources Compile Log Debug Find Results
Line Col Message
In function 'main':
6 1  [Error] expected ';' before '}' token
```

Το πλαίσιο μας πληροφορεί για την ύπαρξη συντακτικού λάθους στην γραμμή 6, πριν το δεξί άγκιστρο. Όντως, πριν από αυτό το άγκιστρο έχουμε ξεχάσει να πληκτρολογήσουμε το ερωτηματικό για να δηλώσουμε το τέλος της εντολής. Υπάρχει περίπτωση, όπως με παλαιότερη έκδοση του λογισμικού, να δούμε πιο γενικά μηνύματα λαθών, όπως απλά ‘Syntax error before “}” token’.

Αμέσως λοιπόν μετά από μία επιτυχή μεταγλώττιση, για να εκτελέσουμε το πρόγραμμά μας, επιλέγουμε “Execute” -> “Run”. Τότε, ανοίγει ένα παράθυρο του κελύφους εντολών των Windows (cmd.exe) στο οποίο περιέχονται οι εκτυπώσεις εκτέλεσης του προγράμματός μας. Αφού το πρόγραμμα ολοκληρώνεται με την εκτέλεση του μηνύματος, υπό κανονικές συνθήκες το παράθυρο

θα έκλεινε αυτόματα και άμεσα. Το Dev-C++ παρέχει τη δυνατότητα εισαγωγής παύσης μετά το τερματισμό του προγράμματος, μέχρι να πατηθεί οποιοδήποτε πλήκτρο.



```
C:\helloworld.exe
Hello world
-----
Process exited with return value 0
Press any key to continue . . .
```

Το Dev-C++ παρέχει πληθώρα διευκολύνσεων, όπως τη δημιουργία “project” για τη διαχείριση πολλαπλών αρχείων πηγαίου κώδικα, οπτική αποσφαλμάτωση του προγράμματός μας, προτάσεις αυτόματης συμπλήρωσης κλπ, τα οποία θα δούμε σε επόμενα εργαστήρια.

2. Η εφαρμογή WinSCP για μεταφορά αρχείων

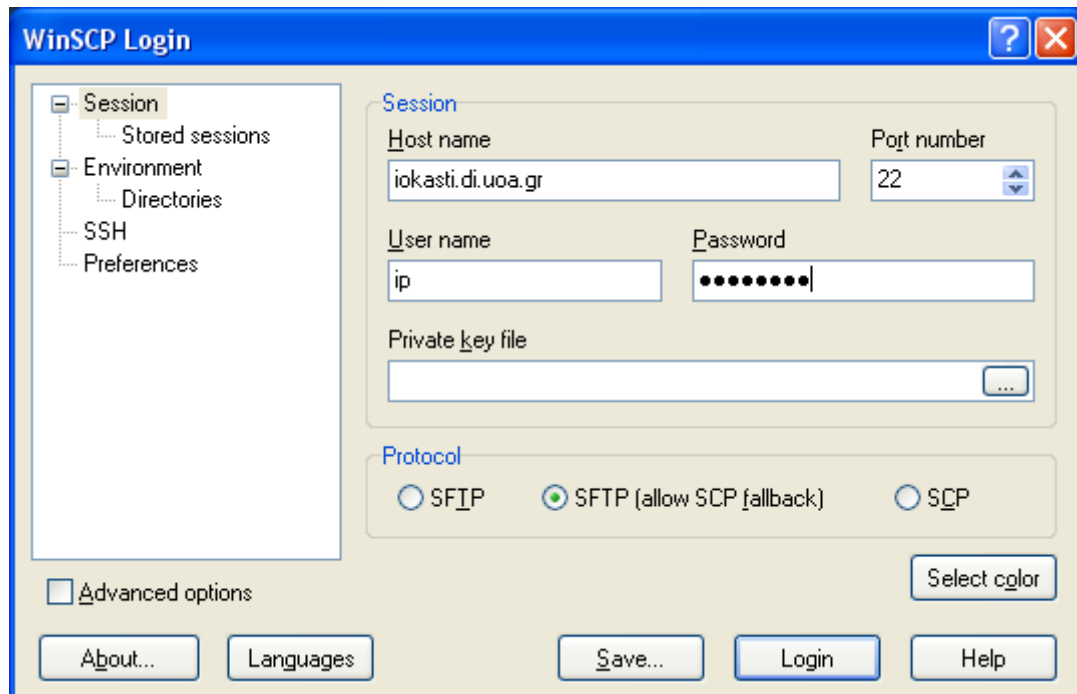
Θα χρησιμοποιήσουμε την εφαρμογή WinSCP για την μεταφορά αρχείων από υπολογιστές Windows στους υπολογιστές των εργαστηρίων UNIX της σχολής. Με τον τρόπο αυτό, μπορούμε να χρησιμοποιούμε τον υπολογιστή μας, ή τους υπολογιστές του εργαστηρίου Windows για να γράφουμε τα προγράμματά μας, να μεταφέρουμε τα αρχεία μας στο Unix και τελικά να ελέγχουμε την ορθή λειτουργία τους με χρήση του μεταγλωττιστή gcc, που είναι και η επίσημη πλατφόρμα εξέτασης του μαθήματος.

Για να εκτελέσουμε το WinSCP, κάνουμε στα εργαστήρια της σχολής Start->Run και πληκτρολογούμε WinSCP.

Από το σπίτι μας, μπορούμε να κατεβάσουμε το πρόγραμμα από την διεύθυνση:

<http://sourceforge.net/projects/winscp/files/latest/download>

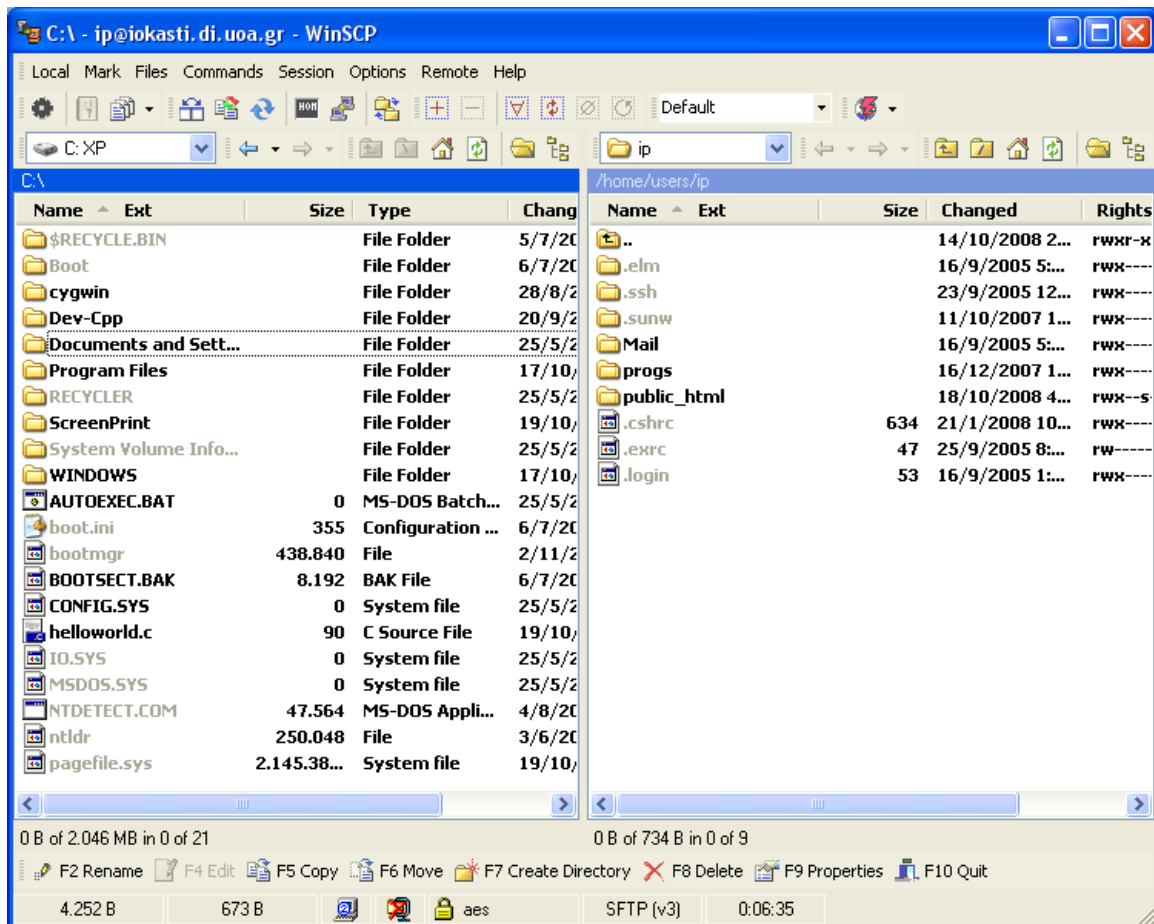
Εκτελώντας το πρόγραμμα βλέπουμε την ακόλουθη οθόνη:



όπου πληκτρολογούμε τα στοιχεία σύνδεσης μας δηλαδή:

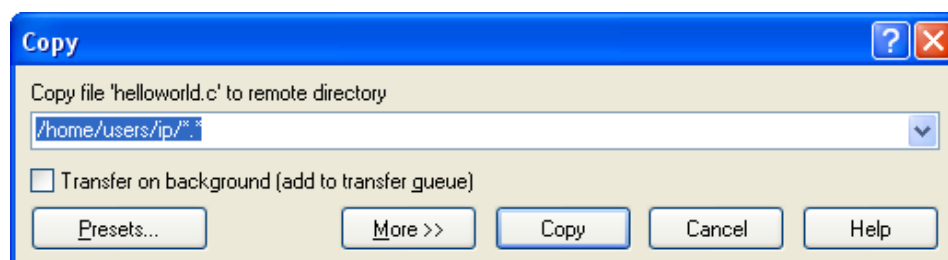
- Τον υπολογιστή που θα συνδεθούμε (http://cgi.di.uoa.gr/~ip/linux_lab_machines.html)
- Το όνομα χρήστη
- Τον κωδικό μας

Και πατάμε το πλήκτρο “Login” οπότε και εμφανίζεται η ακόλουθη οθόνη:



Στο αριστερό μέρος της οθόνης φαίνονται τα περιεχόμενα του τοπικού καταλόγου μας και στο δεξί μέρος της οθόνης φαίνονται τα περιεχόμενα του λογαριασμού μας της σχολής.

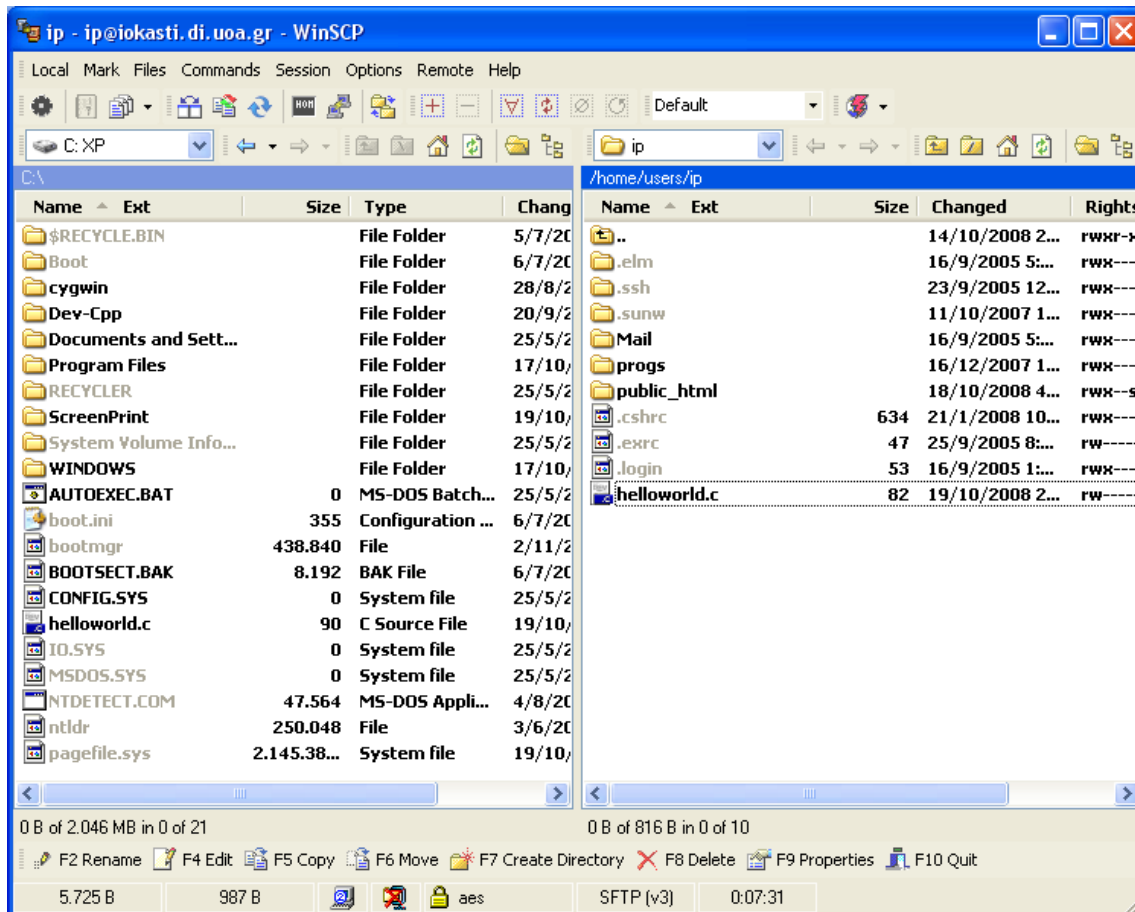
Έτσι, για να μεταφέρουμε αρχεία από τον υπολογιστή μας, στον λογαριασμό της σχολής, επιλέγουμε πρώτα τα αρχεία από το αριστερό μέρος της οθόνης και έπειτα πατάμε το πλήκτρο Copy ή πατάμε το πλήκτρο για συντόμευση F5. Εμφανίζεται τότε το ακόλουθο μήνυμα:



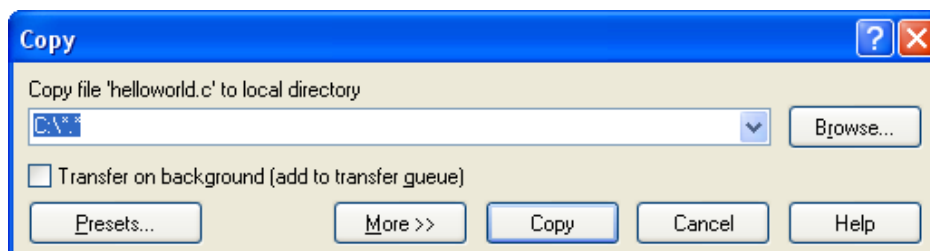
Με αυτό το μήνυμα ζητείται η επιβεβαίωση μας για την μεταφορά του αρχείου από τον τοπικό κατάλογο στον χώρο του λογαριασμού μας της σχολής. Αν πατήσουμε "Copy" το αρχείο μεταφέρεται στον λογαριασμό μας.

Βεβαίως είναι εφικτό να ακολουθήσουμε και την αντίστροφη διαδικασία, για να αντιγράψουμε αρχεία από τον λογαριασμό μας στην σχολή, στον τοπικό δίσκο.

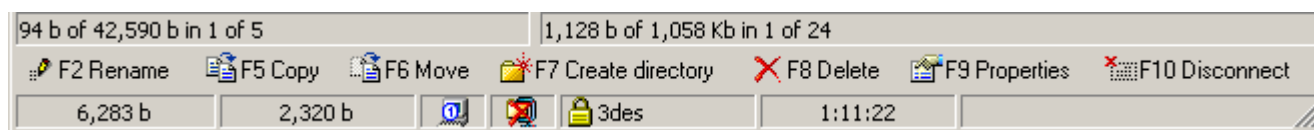
Για να το κάνουμε αυτό επιλέγουμε το αρχείο που μας ενδιαφέρει από το δεξί τμήμα της οθόνης και πατάμε το κουμπί «Copy».



Και πατάμε «Copy» στο επιβεβαιωτικό παράθυρο που εμφανίζεται:



Το WinSCP μας παρέχει και άλλες πρόσθετες δυνατότητες που φαίνονται στο κάτω μέρος της οθόνης:



όπως μετονομασία των αρχείων, δημιουργία καταλόγων, διαγραφή αρχείων και καταλόγων κ.λ.π.

Όταν ολοκληρώσουμε τις εργασίες μας, πατάμε το κουμπί «Disconnect» για να αποσυνδεθούμε.

3. Μεταγλώττιση προγραμμάτων σε περιβάλλον Unix

Κάνουμε login σε περιβάλλον Unix, όπου θα πρέπει να υπάρχει το αρχείο `helloworld.c` που μόλις μεταφέραμε χρησιμοποιώντας το πρόγραμμα WinSCP.

1. Μεταγλωττίστε το αρχείο `helloworld.c` με χρήση του `gcc`, ώστε να παραγάγετε το εκτελέσιμο αρχείο `helloworld`.

2. Εκτελέστε το πρόγραμμα `helloworld`.

Ας ρίξουμε τώρα μια πιο εκτενή ματιά στο πρόγραμμα `helloworld.c`.

```
/* File: helloworld.c */  
  
#include <stdio.h>  
  
main()  
{  
    printf("Hello world\n");  
}
```

Επεξηγήσεις:

Η συνάρτηση `printf()` εμφανίζει στην οθόνη την συμβολοσειρά που δέχεται σαν όρισμα.

Το `'\n'` είναι ο χαρακτήρας αλλαγής γραμμής.

4. Τροποποιείτε την συμβολοσειρά που δέχεται η `printf()` σαν όρισμα, ώστε η έξοδος που εμφανίζεται στην οθόνη να είναι:

```
Hello  
world
```

5. Χρησιμοποιείτε μία δεύτερη `printf()` για να έχετε την εξής έξοδο στο πρόγραμμά σας:

```
Hello  
world  
at  
d.i.t.
```

ΠΑΡΑΡΤΗΜΑ: Αποσφαλμάτωση προγραμμάτων (Πράξη 1^η)

Όταν καλούμαστε να δημιουργήσουμε ένα πρόγραμμα για τη λύση ενός προβλήματος είναι σχεδόν απίθανο να είναι σωστό εξ αρχής. Οποιοσδήποτε, όσο έμπειρος κι αν είναι, θα έχει στο πρόγραμμά του λάθη, τα ονομαζόμενα bugs, τα οποία μπορεί να μην επιτρέπουν τη μεταγλώττιση του προγράμματος ή/και να το κάνουν να μη δουλεύει με τον επιθυμητό τρόπο. Στο σημερινό εργαστήριο θα εστιάσουμε στα συντακτικά λάθη, τα οποία αποτελούν μία μορφή τέτοιων σφαλμάτων, και θα δούμε χρήσιμες τεχνικές για την εύρεση και τη διόρθωσή τους.

Συντακτικά λάθη

Ένα συντακτικό λάθος είναι, όπως υπαγορεύει και το όνομά του, ένα λάθος στη σύνταξη του προγράμματός μας. Για να καταλάβει ο μεταγλωττιστής τα προγράμματά μας πρέπει να είναι γραμμένα με έναν πολύ αυστηρό τρόπο. Οποιαδήποτε παράλειψη σε αυτόν τον αυστηρό τρόπο σύνταξης θα έχει ως αποτέλεσμα την αποτυχία της μεταγλώττισης.

Συχνά συντακτικά λάθη είναι η παράλειψη κάποιας παρένθεσης, το μη κλείσιμο κάποιας αγκύλης, η χρήση μιας μεταβλητής που δεν έχουμε δηλώσει, κλπ. Ας δούμε ένα παράδειγμα:

```
#include <stdio.h>

main() {
    printf("Hello world!\n");
}
```

Αν δώσουμε τον παραπάνω κώδικα προς μεταγλώττιση θα πάρουμε το εξής σφάλμα

```
4  missing terminating " character
5  syntax error before '}' token
```

Αυτό μας πληροφορεί ότι υπάρχει ένα σφάλμα στη γραμμή 4, το οποίο είναι ότι λείπει ένας χαρακτήρας `"`, καθώς και ότι υπάρχει ένα συντακτικό λάθος στη γραμμή 5 πριν το `}`.

Ας δούμε πρώτα το σφάλμα στη γραμμή 4. Με βάση αυτό, έχουμε παραλείψει ένα χαρακτήρα `"`. Όντως, αν το κοιτάξουμε καλά, λείπει το κλείσιμο του `"` στη συμβολοσειρά `Hello world!\n`. Φτιάχνοντας αυτό, αυτόματα φεύγει και το δεύτερο λάθος, το οποίο φανταστήκαμε ότι σχετίζεται με το πρώτο, αφού πριν το `}` στη γραμμή 5 είναι η γραμμή 4, στην οποία ήδη έχουμε υπόψη μας ένα σφάλμα.

Πειραματιστείτε με το παραπάνω πρόγραμμα για να δείτε τα διάφορα συντακτικά λάθη που μπορεί να δημιουργηθούν. Αφαιρέστε το τελικό ερωτηματικό στη συνάρτηση `printf`, αφαιρέστε το `f` απ' το `printf`, γράψτε λάθος το όνομα της `main`, ξεχάστε το `#` στο `include` και ό,τι άλλο σκεφτείτε. Δείτε τα μηνύματα που σας δίνει ο μεταγλωττιστής και προσπαθήστε να καταλάβετε πως σχετίζονται με αυτό που κάνατε. Ήταν όλα τα μηνύματα που σας έβγαλε κατατοπιστικά;

Τεχνικές εύρεσης και διόρθωσης συντακτικών λαθών

Όπως είδαμε και στο προηγούμενο παράδειγμα, τα συντακτικά λάθη είναι εύκολο να τα ανακαλύψουμε αν διαβάσουμε τα μηνύματα του μεταγλωττιστή. Αυτή είναι και η βασική τακτική που χρησιμοποιούμε για να τα εντοπίσουμε και να τα διορθώσουμε. Κάθε μήνυμα του μεταγλωττιστή θα αναφέρει τη γραμμή όπου υπάρχει το συντακτικό λάθος καθώς και μια περιγραφή του. Από αυτά τα δύο στοιχεία μπορούμε, τις περισσότερες φορές, να βρούμε το συντακτικό λάθος.

Ας δούμε όμως, το επόμενο παράδειγμα:

```
#include <stdio.h>

main() {
    printf("Hello world!\n");
```

Αν δώσουμε τον παραπάνω κώδικα προς μεταγλώττιση θα πάρουμε το εξής σφάλμα

```
4 syntax error at end of input
```

Το οποίο μας λέει ότι υπάρχει ένα συντακτικό λάθος στη γραμμή 4 χωρίς καμία επιπλέον υπόδειξη. Το λάθος είναι ότι δεν έχουμε κλείσει την αγκύλη της `main`, οπότε ίσως να αναμέναμε κάτι σαν `"syntax error, bracket needed"`.

Οπότε, είναι εμφανές ότι το να βασιζόμαστε μόνο στα μηνύματα του μεταγλωττιστή για να διορθώσουμε τα συντακτικά λάθη δεν είναι μια τακτική που αποδίδει πάντα. Αυτό που χρειάζεται είναι εμπειρία ώστε να αποκτηθεί εξοικείωση με τα διάφορα συντακτικά λάθη, αλλά και προσοχή κατά τη συγγραφή του κώδικα ώστε να αποφεύγουμε επιπολαιότητες.

