ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΩΝ

ΔΗΜΙΟΥΡΓΙΑ ΕΡΓΑΛΕΙΟΥ ΒΙΟΠΛΗΡΟΦΟΡΙΚΗΣ ΓΙΑ ΤΗΝ ΑΥΤΟΜΑΤΗ ΕΠΕΞΕΡΓΑΣΙΑ ΚΑΙ ΣΥΛΛΟΓΗ ΒΙΒΛΙΟΓΡΑΦΙΚΩΝ ΠΛΗΡΟΦΟΡΙΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΦΑΣΜΑΤΟΓΡΑΦΙΑΣ ΜΑΖΑΣ

Διπλωματική Εργασία

ΣΑΚΚΟΣ ΝΙΚΟΛΑΟΣ

Επιβλέπων: Νικόλαος Ουζούνογλου
Καθηγητής ΕΜΠ

ΑΘΗΝΑ, ΝΟΕΜΒΡΙΟΣ **2007**

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΩΝ

ΔΗΜΙΟΥΡΓΙΑ ΕΡΓΑΛΕΙΟΥ ΒΙΟΠΛΗΡΟΦΟΡΙΚΗΣ ΓΙΑ ΤΗΝ ΑΥΤΟΜΑΤΗ ΕΠΕΞΕΡΓΑΣΙΑ ΚΑΙ ΣΥΛΛΟΓΗ ΒΙΒΛΙΟΓΡΑΦΙΚΩΝ ΠΛΗΡΟΦΟΡΙΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΦΑΣΜΑΤΟΓΡΑΦΙΑΣ ΜΑΖΑΣ

Διπλωματική Εργασία

ΣΑΚΚΟΣ ΝΙΚΟΛΑΟΣ

Επιβλέπων: Νικόλαος Ουζούνογλου
Καθηγητής ΕΜΠ

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 30η Νοεμβρίου 2007.

…….....	…….....	…….....
Π. Φράγκος Καθηγητής Ε.Μ.Π.	Ν. Ουζούνογλου Καθηγητής Ε.Μ.Π.	Η. Αβραμόπουλος Αν. Καθηγητής Ε.Μ.Π.

ΑΘΗΝΑ, ΝΟΕΜΒΡΙΟΣ 2007

Νικόλαος Σάκκος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περιληψη

Καθώς τα βιολογικά δεδομένα πολλαπλασιάζονται συνεχώς, έγινε επιτακτική η χρήση μεθόδων ανάλυσης δεδομένων σε μεγάλη κλίμακα. Όσον αφορά τα πρωτεϊνικά δεδομένα, η τεχνολογία που με την ανάπτυξή της επέτρεψε στους ερευνητές να επεξεργάζονται μεγάλους όγκους πληροφορίας είναι η φασματογραφία μάζας. Με τη χρήση των φασματογράφων μάζας μπορούμε να προσδιορίσουμε επακριβώς τις υπάρχουσες πρωτεΐνες σε ένα άγνωστο δείγμα.

Σκοπός αυτής της διπλωματικής εργασίας είναι η δημιουργία μιας εφαρμογής, που επεξεργάζεται αυτόματα τα αποτελέσματα φασματογραφίας μάζας και εξάγει βιβλιογραφικές πληροφορίες για τις πρωτεΐνες που βρέθηκαν από βιολογικές βάσεις δεδομένων και συγκεκριμένα τις «UniProt», «InterPro», «AmiGO!».

Η αυτόματη συλλογή πληροφοριών είναι πολύ σημαντική, δεδομένου του ότι είναι πολύ χρονοβόρα αν γίνεται για κάθε μία από τις πρωτεΐνες ξεχωριστά.

Η υλοποίηση της εφαρμογής έγινε με τη χρήση της γλώσσας Perl ενώ χρησιμοποιήθηκε το MySQL Σύστημα Διαχείρισης Βάσης Δεδομένων κι ο Apache Web Server.

Λεξεις Κλειδια

βιοπληροφορική, πρωτεωμική, πρωτεΐνη, φασματογραφία μάζας, UniProt, βιολογική βάση δεδομένων, βιβλιογραφικές πληροφορίες, Perl

ABSTRACT

As biological data have been growing exponentially in recent years, the use of high-throughput analysis methods has become essential to biological research. The high throughput technology that has given proteomic research a significant boost is mass spectrometry. With the use of mass spectrometers, unknown proteins within a mixture can be identified.

The scope of this thesis was the development of an application that analyzes mass spectrometry results and automatically extracts bibliographical information about the proteins found within the sample, using biological databases like UniProt, InterPro, AmiGO.

The automatic extraction of bibliographical information is very useful, considered that manual searches on hundreds of proteins is very time consuming.

The application was written in the Perl programming language, which is widely used in bioinformatics. The DBMS used was MySQL and the web server used was Apache.

KEYWORDS

bioinformatics, proteomics, protein, mass spectrometry, biological database, bibliographical information, Perl

Ε Υ Χ Α Ρ Ι Σ Τ Ι Ε Σ

Ευχαριστώ το Γιώργο Λούντο, τη Σοφία Κοσσίδα, και την Ελένη Κατσαντώνη, για όλη τη βοήθεια που προσέφεραν κατά την εκπόνηση αυτής της διπλωματικής.

Η εργασία αυτή πραγματοποιήθηκε στο Τμήμα Αιματολογίας-Ογκολογίας του Ιδρύματος Ιατροβιολογικών Ερευνών, Ακαδημίας Αθηνών, υπό την επίβλεψη της ερευνήτριας Ελένης Κατσαντώνη. Τα δεδομένα φασματογραφίας μάζας που χρησιμοποιήθηκαν για τη δημιουργία και τον έλεγχο της εφαρμογής δημιουργήθηκαν από τη διδακτορική φοιτήτρια Βασιλική Λάζου και την Ελένη Κατσαντώνη.

1. ΕΙΣΑΓΩΓΗ

1.1 Τι είναι η Βιοπληροφορική

Τα βιολογικά δεδομένα πολλαπλασιάζονται συνεχώς. Βάσεις δεδομένων που είναι ανοιχτές στο κοινό, όπως η GenBank και Protein Data Bank αναπτύσσονται εκθετικά εδώ και αρκετό καιρό. Με την έλευση του Παγκόσμιου Ιστού και των γρήγορων συνδέσεων διαδικτύου, τα δεδομένα που περιέχονται σε αυτές τις βάσεις δεδομένων και σε εξειδικευμένες εφαρμογές μπορούν να χρησιμοποιηθούν γρήγορα, εύκολα και χωρίς κόστος από οποιοδήποτε μέρος του κόσμου. Ως συνέπεια, τα εργαλεία πληροφορικής διαδραματίζουν πλέον πολύ σημαντικό ρόλο στην ανάπτυξη και πρόοδο της βιολογικής έρευνας.

Η βιοπληροφορική, ένας ραγδαία εξελισσόμενος επιστημονικός κλάδος, είναι η εφαρμογή των υπολογιστικών εργαλείων και τεχνικών στη διαχείριση και ανάλυση των βιολογικών δεδομένων.

1.2 -omics

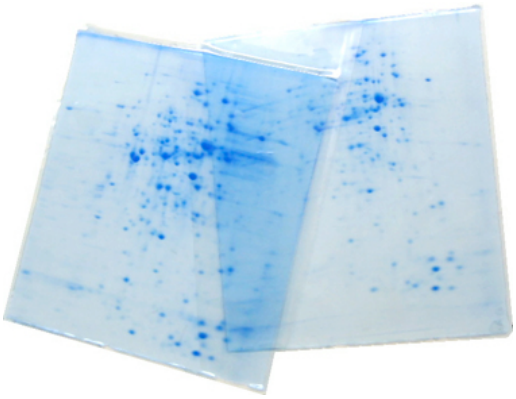
Τα τελευταία χρόνια, η βιολογική έρευνα έχει γνωρίσει εκρηκτική ανάπτυξη με την εισαγωγή νέων τεχνολογιών που επιτρέπουν την γρήγορη επεξεργασία μεγάλου όγκου δεδομένων (high-throughput analysis). Αυτή η εξέλιξη της τεχνολογίας έδωσε ώθηση σε νέα πεδία επιστημών που μοιράζονται τη δημοφιλή κατάληξη -omics. Πολλοί διαφορετικοί όροι χρησιμοποιούνται για να κατηγοριοποιήσουν τα νέα αντικείμενα μελέτης τα οποία και συνεχώς αυξάνονται. Μερικά από αυτά είναι

- Genomics, που είναι η μελέτη του συνόλου των γονιδίων ενός οργανισμού
- Proteomics, που είναι η μελέτη των πρωτεϊνών, κυρίως των δομών και των λειτουργιών τους
- Metabolomics, που είναι η μελέτη των μοναδικών χημικών αποτυπωμάτων που συγκεκριμένοι κυτταρικοί μηχανισμοί αφήνουν ως κατάλοιπα.
- Interactomics, που είναι η μελέτη του συνόλου των μοριακών αλληλεπιδράσεων στα κύτταρα

1.3 Proteomics-Mass Spectrometry

Πρωτεωμική (Proteomics) είναι η μεγάλης κλίμακας μελέτη των πρωτεϊνών, ειδικά των δομών και των λειτουργιών τους, ενώ το πρωτέωμα (proteome) ενός οργανισμού αντιστοιχεί στο σύνολο των πρωτεϊνών που ενυπάρχουν σε έναν οργανισμό καθ' όλη τη διάρκεια της ζωής του. Συνεπώς, η πρωτεωμική ασχολείται με τη μελέτη της σύνθεσης, δομής, λειτουργίας και αλληλεπίδρασης των πρωτεϊνών, οι οποίες διευθύνουν τις δραστηριότητες κάθε ζωντανού κυττάρου.

Η σημερινή έρευνα στην πρωτεωμική απαιτεί πρώτα οι πρωτεΐνες να διασπαστούν, πολλές φορές σε μεγάλη κλίμακα. Ο διαχωρισμός των πρωτεϊνών μπορεί να γίνει χρησιμοποιώντας ηλεκτροφόρηση δισδιάστατης γέλης (2-D gel electrophoresis), η οποία συνήθως διαχωρίζει τις πρωτεΐνες πρώτα σύμφωνα με το ισοηλεκτρικό τους σημείο και μετά σύμφωνα με το μοριακό τους βάρος. Οι πρωτεϊνικές κηλίδες μπορούν να οπτικοποιηθούν με χρήση διάφορων χημικών χρωστικών και πολλές φορές μπορούν να προσδιοριστούν ποσοτικά από την ένταση του αποτυπώματος (intensity of the stain). Αφού οι πρωτεΐνες διαχωρισθούν και προσδιοριστούν ποσοτικά, μπορούν να αναγνωρισθούν.



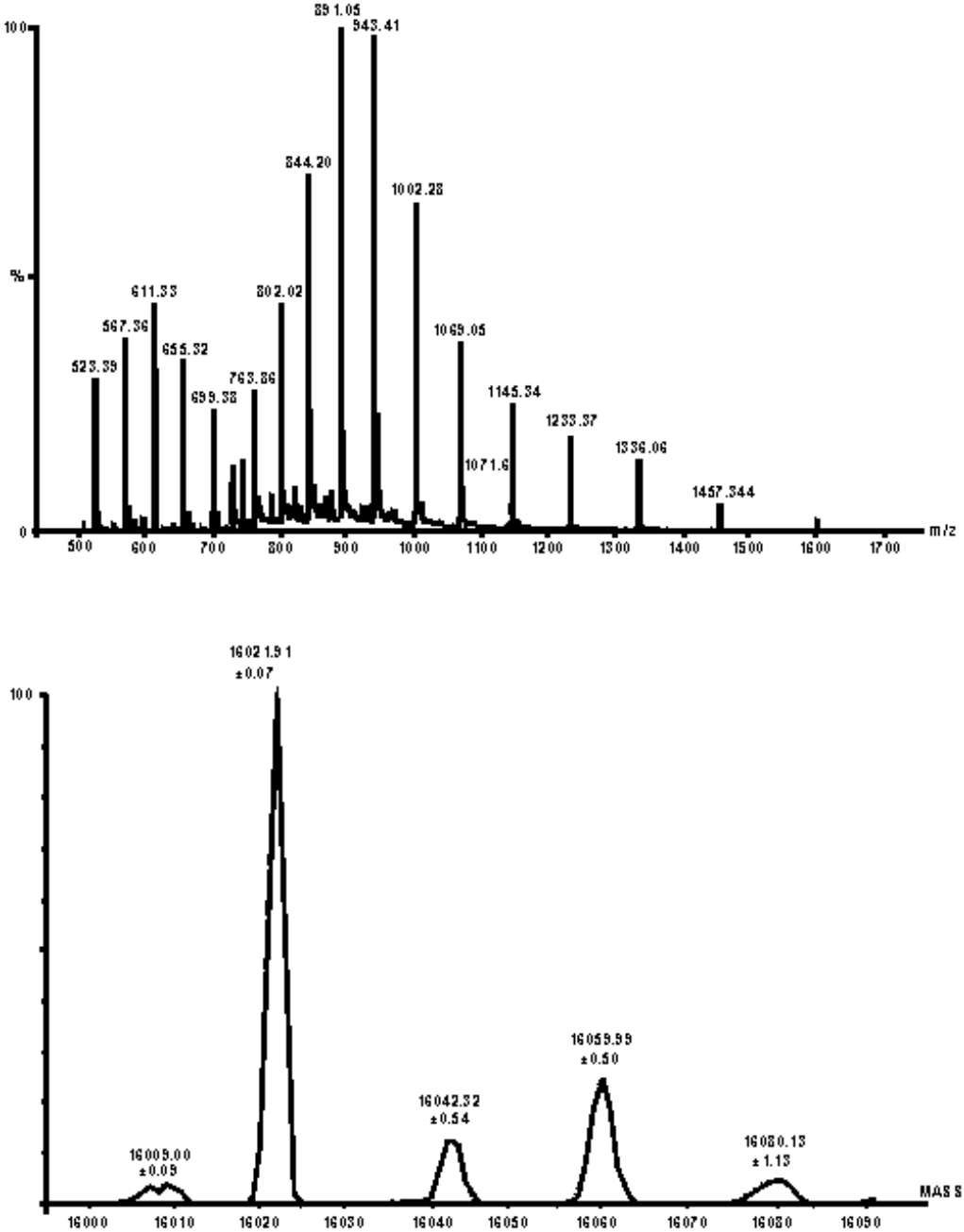
Coomassie stained 2D gels

Κάθε κηλίδα αποκόβεται από τη γέλη και τεμαχίζεται σε πεπτιδικές αλυσίδες με τη χρήση πρωτεολυτικών ενζύμων (proteolytic enzymes). Αυτές οι πεπτιδικές αλυσίδες μπορούν να αναγνωριστούν με τη χρήση φασματογραφίας μάζας και συγκεκριμένα MALDI-TOF (matrix-assisted laser desorption-ionization time of flight mass spectrometry) φασματογραφία μάζας.

Σε αυτή τη μέθοδο, μια πεπτιδική αλυσίδα ιονίζεται με τη χρήση μιας ακτίνας λέιζερ και μια αύξηση στην τάση του πλέγματος χρησιμοποιείται για να εξαπολύσει τα ιόντα προς ένα ανιχνευτή στον οποίο ο χρόνος που χρειάζεται ένα ιόν για να φτάσει τον ανιχνευτή εξαρτάται από τη μάζα του. Όσο μεγαλύτερη η μάζα, τόσο μεγαλύτερος κι ο χρόνος που χρειάζεται το ιόν για να φτάσει στον ανιχνευτή (time of flight). Σε ένα φασματογράφο μάζας MALDI-TOF, τα ιόντα μπορούν εκτραπούν με τη βοήθεια ενός ηλεκτροστατικού ανακλαστήρα, ο οποίος μπορεί επιπρόσθετα να εστιάζει την ιοντική ακτίνα. Συνεπώς, οι μάζες των ιόντων που φτάνουν στον δεύτερο ανιχνευτή

μπορούν να προσδιορισθούν με μεγάλη ακρίβεια κι αυτές οι μάζες μπορού να αποκαλύψουν την ακριβή χημική σύνθεση των πεπτιδικών αλυσίδων και συνεπώς να τις αναγνωρίσουν.

Αποτελέσματα φασματογραφίας μάζας



His-Tagged Human b5

1.4 Το πρόβλημα και η λύση του

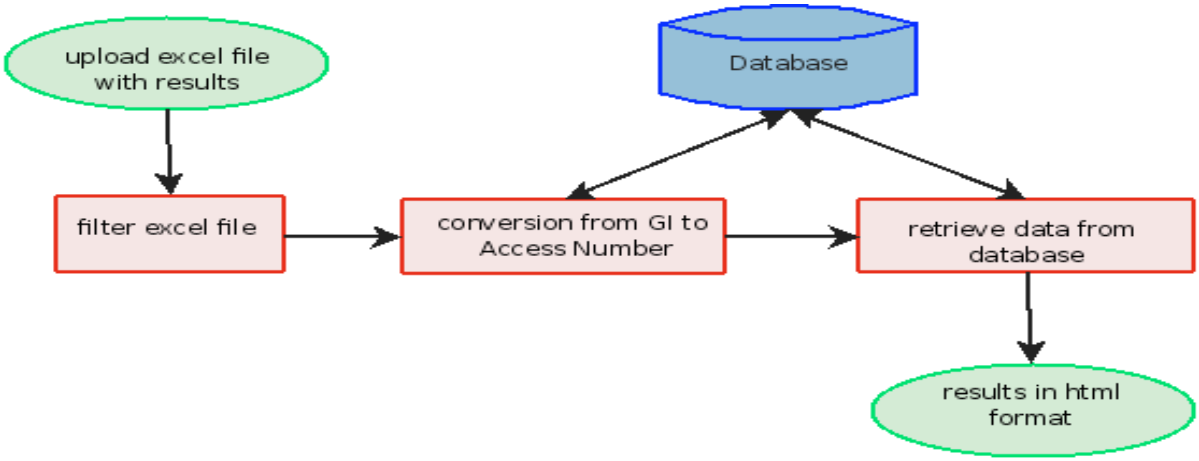
Σκοπός της εργασίας αυτής, είναι η δημιουργία ενός εργαλείου που θα συλλέγει και θα επεξεργάζεται αυτόματα βιβλιογραφικές πληροφορίες αποτελέσματων φασματογραφίας μάζας. Τα αποτελέσματα φασματογραφίας μάζας παρείχε η Έλενα Κατσαντώνη, ερευνήτρια στο Τμήμα Αιματολογίας-Ογκολογίας του Ιδρύματος Ιατροβιολογικών Ερευνών Ακαδημίας Αθηνών (ΙΙΒΕΑΑ) και η διδακτορική φοιτήτρια Βασιλική Λάζου. Τα συγκεκριμένα αποτελέσματα φασματογραφίας μάζας έχουν σχέση με απομόνωση πρωτεϊνικών συμπλεγμάτων του STAT5.

Ο παράγοντας STAT5, εμπλέκεται σε μηχανισμούς μεταγωγής σήματος από το κυτταρόπλασμα στον πυρήνα του κυττάρου. Συνεχής ενεργοποίηση του STAT5 είναι χαρακτηριστική σε πολλούς τύπους καρκίνου και στοχευμένη απενεργοποίησή τους θα αποτρέψει τον καρκινικό μετασχηματισμό. Για το λόγο αυτό είναι σημαντική η διερεύνηση της μοριακής δράσης του STAT5, τόσο σε επίπεδο αλληλεπιδράσεων με άλλες πρωτεΐνες, όσο και σε επίπεδο των γονιδίων που ρυθμίζει.

Εφαρμόστηκε μεθοδολογία in vivo βιοτινυλίωσης (de Boer et al, 2003) για τον προσδιορισμό πρωτεϊνικών συμπλεγμάτων του STAT5. Χρησιμοποιώντας τη μεθοδολογία αυτή απομονώθηκαν πιθανές αλληλεπιδρούσες με το STAT5 πρωτεΐνες με φασματομετρία μάζας (NanoLC-MS/MS). Οι πρωτεΐνες αυτές είναι εκατοντάδες και το εργαλείο δημιουργήθηκε για να γίνει *αυτόματη συλλογή πληροφοριών για τις διάφορες πρωτεΐνες από τις βάσεις δεδομένων UniProt, InterPro, AmiGO.*

Η αυτόματη συλλογή πληροφοριών είναι πολύ σημαντική, δεδομένου του ότι είναι πολύ χρονοβόρα αν γίνεται για κάθε μία από τις πρωτεΐνες ξεχωριστά.

Για την επίλυση του προβλήματος, εγκαταστήσαμε τοπικά μια βάση δεδομένων, η οποία περιέχει το σύνολο των πληροφοριών που χρειαζόμαστε. Ο χρήστης έχει τη δυνατότητα να ανανεώνει χειροκίνητα αυτή τη βάση δεδομένων, όταν υπάρχουν ανανεώσεις των επιμέρους βάσεων δεδομένων. Τα δεδομένα εισόδου είναι αναγνωριστικοί αριθμοί (id numbers) των πρωτεϊνών που ενδιαφέρουν το χρήστη. Τα δεδομένα εξόδου είναι πληροφορίες που προκύπτουν για αυτές τις πρωτεΐνες από τη σχετική βιβλιογραφία, ώστε να δώσουν στο χρήστη συνοπτικές πληροφορίες για τη λειτουργικότητά τους. Ακολουθεί διάγραμμα που εξηγεί τη βασική ροή του προγράμματος


upload excel file with results
Database
filter excel file
conversion from GI to Access Number
retrieve data from database
results in html format

2.ΥΛΟΠΟΙΗΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

2.1 Εργαλεία Πληροφορικής που χρησιμοποιήθηκαν

PERL

Για την υλοποίηση του εργαλείου χρησιμοποιήθηκε η γλώσσα Perl. Οι λόγοι που μας ώθησαν σε αυτή την επιλογή είναι

- το γεγονός πως η Perl χρησιμοποιείται ευρέως στον τομέα της βιοπληροφορικής. Υπάρχουν ήδη πολλά εργαλεία γραμμένα σε Perl (Perl modules) από τα οποία μπορούμε να αντλήσουμε γνώσεις ή και να τα επαναχρησιμοποιήσουμε εφόσον εξυπηρετούν το σκοπό μας (BioPerl).
- Στον τομέα της βιοπληροφορικής οι πληροφορίες είναι συνήθως αποθηκευμένες σε ογκώδεις βάσεις δεδομένων και αρχεία κειμένου μεγέθους δεκάδων κι εκατοντάδων GigaByte (flat files). Η Perl είναι γλώσσα ειδικά σχεδιασμένη και ιδιαίτερα αποδοτική στην αναγνώριση προτύπων (pattern matching) και στην αναζήτηση ακολουθιών χαρακτήρων (strings) σε μεγάλο όγκο δεδομένων, όπως στις παραπάνω περιπτώσεις.
- Είναι γλώσσα υψηλού επιπέδου και αποκρύπτει από το χρήστη λειτουργίες χαμηλότερου επιπέδου όπως η διαχείριση της μνήμης. Πολύ συχνά χρειάζεται λιγότερος κώδικας σε Perl για την επίλυση ενός προβλήματος σε σχέση με C ή Java.
- Είναι ανεξάρτητη της πλατφόρμας ανάπτυξης (platform independent). Ο ίδιος κώδικας μπορεί να επαναχρησιμοποιηθεί σε Microsoft Windows, Linux, Mac OS και σε οποιοδήποτε σύστημα είναι βασισμένο σε Unix.

MYSQL

Είναι το πιο δημοφιλές Σύστημα Διαχείρισης Βάσεων Δεδομένων ανοιχτού κώδικα. Χρησιμοποιήσαμε αυτή τη βάση δεδομένων για την καταχώρηση των πληροφοριών που πρέπει να διαχειριστούμε. Ως διεπαφή μεταξύ του κυρίως προγράμματος και της βάσεως δεδομένων, χρησιμοποιήθηκε το Perl DataBase Interface (DBI module).

Apache Web Server

Εξυπηρετητής παγκόσμιου ιστού ανοικτού κώδικα. Χρησιμοποιήθηκε για την υλοποίηση της διεπαφής του εργαλείου με το χρήστη. Η διεπαφή είναι μια ιστοσελίδα μέσω της οποίας ο χρήστης δίνει τις πληροφορίες εισόδου. Στη συνέχεια τα αποτελέσματα παρουσιάζονται δυναμικά, ως αρχεία html τα οποία ο χρήστης μπορεί να δει στη συνέχεια μέσω ενός web browser. Για τη δημιουργία των δυναμικών σελίδων χρησιμοποιήθηκε το Perl CGI module (Common Gateway Interface).

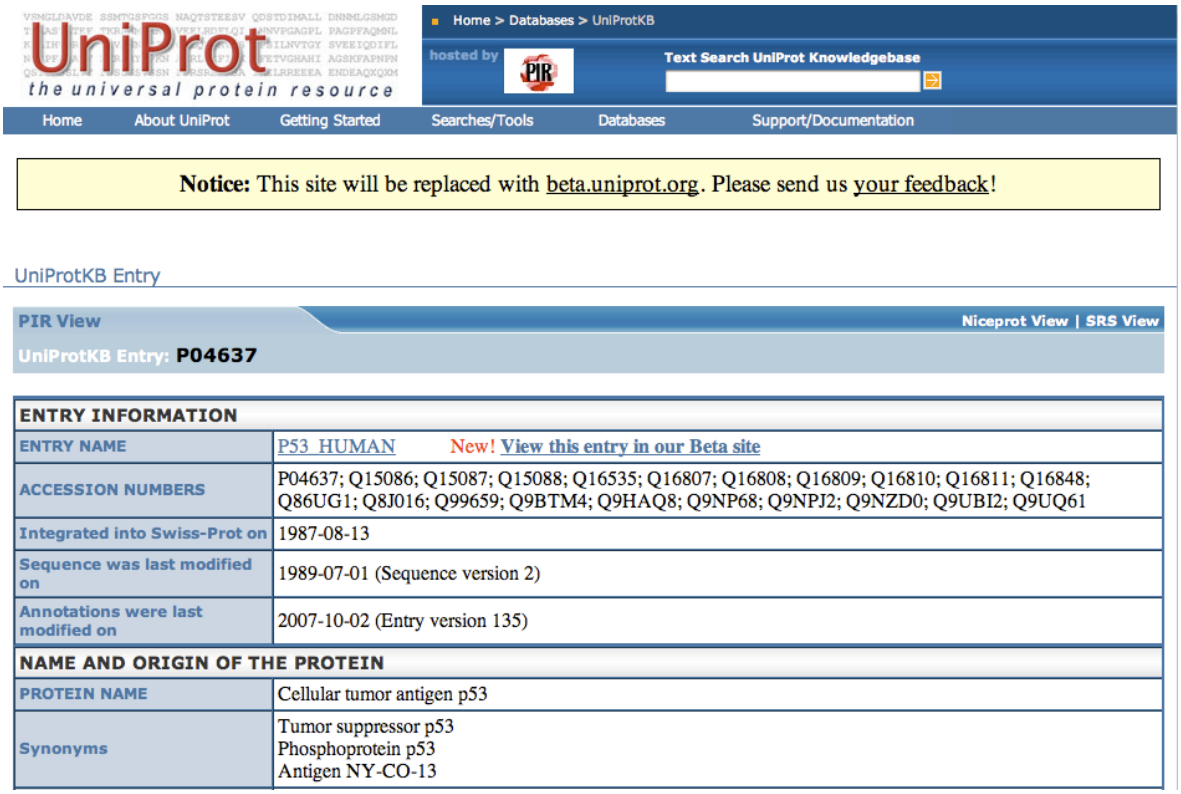
2.2 Βιολογικές Βάσεις δεδομένων που χρησιμοποιήθηκαν

Οι πληροφορίες που μας ενδιαφέρουν αντλούνται από τρεις διαφορετικές βάσεις δεδομένων (*UniProt, AmiGO, InterPro*). Όλες οι παραπάνω βάσεις παρέχουν στον ερευνητή διαδικτυακά εργαλεία για την πρόσβαση στις πληροφορίες που διαθέτουν. Παρόλα αυτά, θέτουν περιορισμούς στο πλήθος των αναζητήσεων που μπορεί να κάνει ένας χρήστης και στο χρόνο που μπορεί να μεσολαβήσει ανάμεσα σε αυτές, για την εξοικονόμηση και ισοκατανομή των πόρων ανάμεσα σε διαφορετικούς χρήστες. Για αυτό το λόγο προτιμήθηκε μια τοπική εγκατάσταση των βάσεων από την πρόσβαση σε αυτές μέσω του διαδικτύου.

Ακολουθούν περισσότερες πληροφορίες για τις επιμέρους βάσεις δεδομένων

UNIPROT

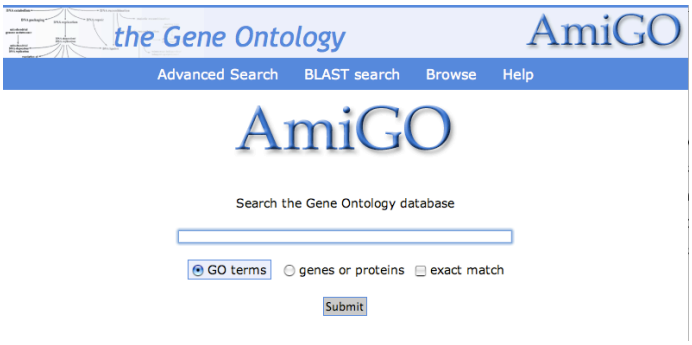
http://www.ebi.uniprot.org/



Το UniProt είναι η παγκόσμια βάση δεδομένων για πρωτεΐνες (Universal Protein database), η οποία δημιουργήθηκε από την ένωση των βάσεων Swiss-Prot, TrEMBL και PIR. Αυτό την καθιστά την πιο εκτενή και περιεκτική πηγή πληροφοριών σε σχέση με πρωτεΐνες.

Το UniProt είναι αποτέλεσμα της κοινοπραξίας των EBI (European BioInformatics Institute), SIB (Swiss Institute of Bionformatics) και PIR (Protein Information Resource).

AMIGO

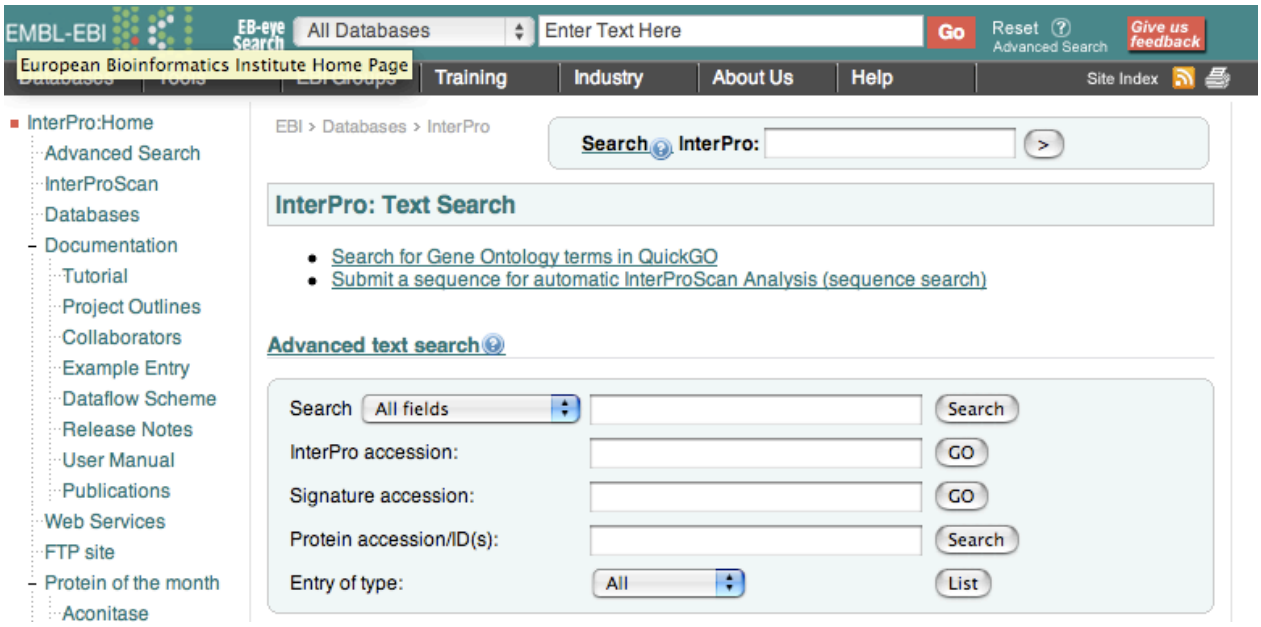


http://amigo.geneontology.org/

Το Gene Ontology Project παρέχει πληροφορίες σχετικά με τις ιδιότητες των γονιδίων και των προϊόντων τους. Μας ενδιαφέρουν οι πληροφορίες που παρέχει για τη λειτουργικότητα των γονιδίων και των προϊόντων τους (molecular function, biological processes, cellular components). Οι πληροφορίες ανανεώνονται σε μηνιαία βάση. Από το τέλος του 2005, περιέχει πάνω από 19.000 καταχωρήσεις κι αποτελεί πολύτιμο εργαλείο της βιοπληροφορικής.

INTERPRO

http://www.ebi.ac.uk/interpro/

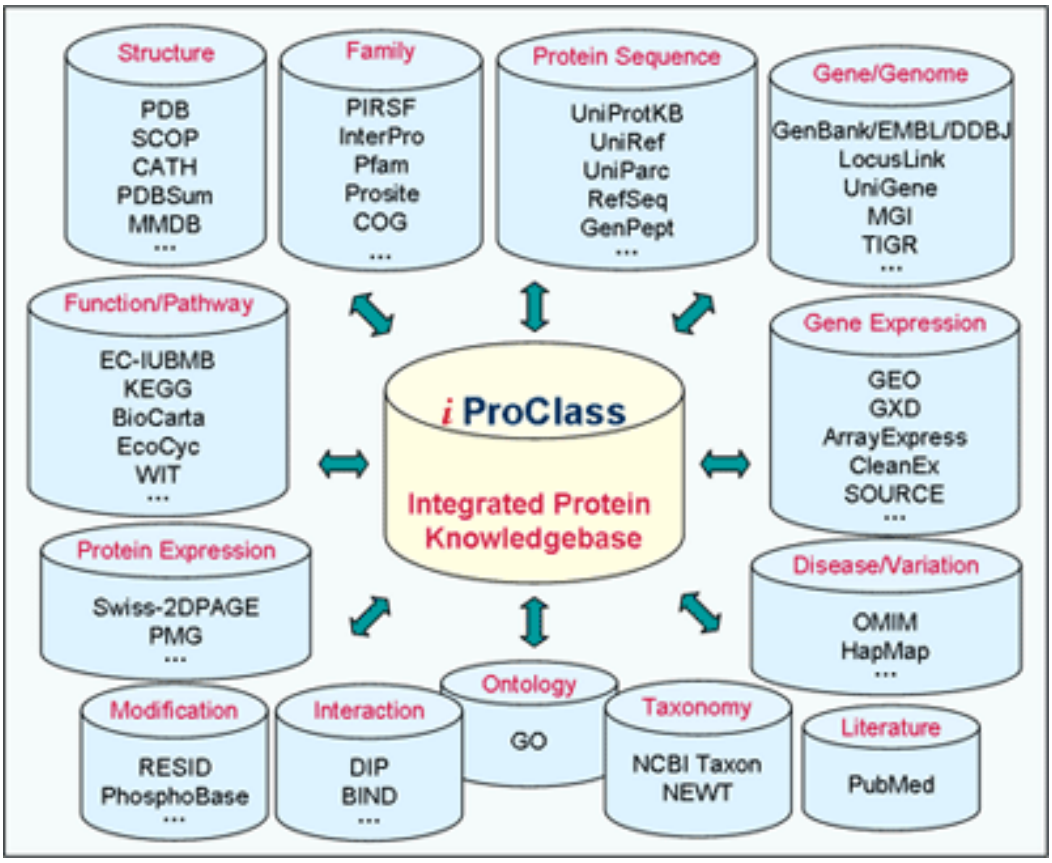


Το InterPro είναι μια βάση δεδομένων με πληροφορίες για τις οικογένειες των πρωτεϊνών, όπου καταγεγραμμένα χαρακτηριστικά γνωστών πρωτεϊνών μπορούν να μας δώσουν πληροφορίες για άγνωστες πρωτεϊνικές ακολουθίες.

IPROCLASS

http://pir.georgetown.edu/iproclass/

Η iProClass είναι μια συγκεντρωτική βάση δεδομένων, που περιέχει 90 βάσεις βιολογικών δεδομένων, μεταξύ των οποίων βρίσκονται και οι βάσεις πληροφοριών που μας ενδιαφέρουν.



Χρησιμοποιήθηκε αντί των υπόλοιπων ξεχωριστών βάσεων για λόγους επεκτασιμότητας. Με τη χρήση της iProClass μπορούμε να προσθέσουμε και νέες βάσεις, αν αυτό κριθεί απαραίτητο στο μέλλον, με μικρές αλλαγές στον κώδικα της εφαρμογής.

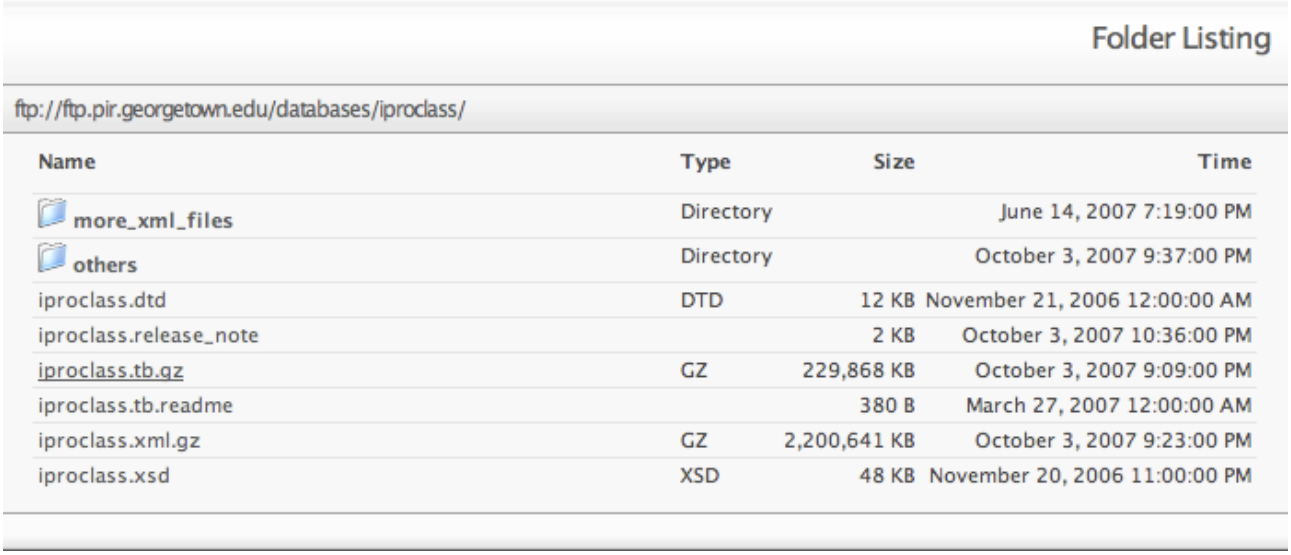
2.3 Προετοιμασία της βάσης δεδομένων

2.3.1 Το περιεχόμενο της βάσης δεδομένων

Πριν χρησιμοποιήσουμε την εφαρμογή, πρέπει να αρχικοποιήσουμε τη βάση δεδομένων. Για να γίνει αυτό, χρειαζόμαστε δυο αρχεία από την ιστοσελίδα του iProClass, τα οποία περιέχουν το σύνολο της πληροφορίας που χρειαζόμαστε.

Τα αρχεία αυτά είναι διαθέσιμα από τη διεύθυνση

ftp://ftp.pir.georgetown.edu/databases/iproclass/



Το πρώτο αρχείο είναι το **iproclass.tb.gz** και το δεύτερο βρίσκεται στο φάκελο **more_xml_files** και έχει την ονομασία **m_musculus.xml.gz**.

I P R O C L A S S . T B

Το *iproclass.tb* είναι ένα απλό αρχείο κειμένου. Περιέχει ένα πίνακα με τα id numbers (μοναδικά αναγνωριστικά) των πρωτεϊνών όλων των οργανισμών που υπάρχουν στη βάση δεδομένων του iProClass. Ο πίνακας αυτός είναι απαραίτητος για τη μετατροπή του αναγνωριστικού αριθμού μιας βάσης στο αναγνωριστικό μιας άλλης.

Συγκεκριμένα, μας επιτρέπει την αντιστοίχηση μεταξύ 22 διαφορετικών αναγνωριστικών αριθμών. Κάθε γραμμή του πίνακα αντιστοιχεί σε μια πρωτεΐνη, κάθε στήλη σε διαφορετικό αναγνωριστικό αριθμό.

Στην εφαρμογή μας, μας ενδιαφέρει περισσότερο η αντιστοίχηση μεταξύ των αναγνωριστικών GI (GenInfo Identifier) και UniProt Accession Number. Ακολουθεί μια σύντομη περιγραφή των δυο αναγνωριστικών αριθμών.

- **GI number**: Προκύπτει από τη βάση δεδομένων του NCBI (National Center for Biotechnology Information) η οποία ανήκει στο NLM (United States National Library of Medicine). Οι αριθμοί αυτοί είναι μια σειρά ψηφίων που ανατίθενται διαδοχικά σε κάθε ακολουθία που καταχωρείται από το NCBI.
- **UniProt Accession Number:** Αναγνωριστικός αριθμός που προκύπτει από τη βάση δεδομένων του UniProt-SwissProt.

M _ M U S C U L U S . X M L

Περιέχει όλες τις πληροφορίες που διαθέτει η βάση δεδομένων του iProClass για τον οργανισμό «*mus musculus*», δηλαδή το κοινό ποντίκι το οποίο και μας ενδιαφέρει. Διατίθεται σε μορφή αρχείου XML (Extensible Markup Language).

2.3.2 Εισαγωγή των δεδομένων στη βάση με χρήση βοηθητικών script

Έχουμε πλέον αυτά τα δυο αρχεία, iproclass.tb και m_musculus.xml, όμως δεν μπορούμε ακόμα να τα αξιοποιήσουμε. Η αναζήτηση λέξεων σε αρχεία κειμένου εκατοντάδων MB είναι χρονοβόρα, ειδικά στην περίπτωση που θέλουμε να κάνουμε πολλαπλές αναζητήσεις. Μένει να εισάγουμε αυτά

τα δεδομένα σε μια τοπικά εγκατεστημένη βάση δεδομένων. Αυτό θα γίνει με τη βοήθεια δυο βοηθητικών script γραμμένων σε perl.

1. **gi2an.pl:** αναλαμβάνει να εισάγει τις πληροφορίες του αρχείου **iproclass.tb** στη βάση δεδομένων, δηλαδή τις στήλες που έχουν τα **GI number** και **Accession Number**
2. **update_iproclass.pl:** αναλαμβάνει να εισάγει τις πληροφορίες του αρχείου **mus_musculus.xml** στη βάση δεδομένων.

Πιο αναλυτικά

G I 2 A N . P L

Το αρχείο iproclass.tb είναι αρχείο ASCII κειμένου, όπου κάθε γραμμή αντιστοιχεί σε μια πρωτεΐνη, κάθε στήλη σε ένα ξεχωριστό identifier. Συγκεκριμένα οι στήλες αντιστοιχούν σε:

1. UniProt_ac (or UniParc_ac with taxon_id)
2. UniProt_id
3. EntrezGene
4. RefSeq
5. GIID
6. PDB
7. PFAM
8. GO
9. PIRSF
10. IPI
11. UniRef_100
12. UniRef_90
13. UniRef_50
14. UniParc

15. PIR-PSD

16. Taxon ID

17. OMIM

18. UniGene

19. Ensemble ID

20. PMID

21. EMBL DNA AC

22. EMBL Protein AC

Χρησιμοποιείται ως εξής:

USAGE: perl gi2an.pl <file> <dbuser> <dbpassword>

όπου

- file: το αρχείο iproclass.tb
- dbuser: το όνομα του χρήστη της βάσης δεδομένων που έχει δικαιώματα να είσαγει δεδομένα
- dbpassword: ο κωδικός του χρήστη
- dbname: το όνομα της βάσης στην οποία θα καταχωρηθούν τα δεδομένα

Παίρνει τις στήλες Uniprot_ac και GIID και τις καταχωρεί σε ένα ξεχωριστό πίνακα στη βάση δεδομένων. Αν ο πίνακας υπάρχει ήδη από προηγούμενη εγκατάσταση, σβήνεται έτσι ώστε να καταχωρηθεί η καινούρια έκδοσή του. Οι γραμμές του πίνακα είναι ταξινομημένες σύμφωνα με τον αναγνωριστικό αριθμό GI.

Για την επικοινωνία με τη βάση δεδομένων, χρησιμοποιήθηκε το DBI module της Perl. Το DBI module είναι χρησιμοποιείται ως διεπαφή της Perl για βάσεις δεδομένων. Ορίζει ένα σύνολο μεθόδων και μεταβλητών που μπορούν να χρησιμοποιηθούν ανεξάρτητα από τη βάση που πραγματικά χρησιμοποιείται (είτε αυτή είναι η MySQL, PostgreSQL ή Oracle). Το ίδιο module χρησιμοποιείται κι οπουδήποτε αλλού στην εφαρμογή χρειάζεται επικοινωνία με τη βάση δεδομένων.

U P D A T E _ I P R O C L A S S . P L

Το script αυτό αναλαμβάνει να εξάγει πληροφορίες από το αρχείο mus_musculus.xml και να τις καταχωρήσει στη βάση δεδομένων. Το αρχείο αυτό είναι της μορφής XML.

Το script χρησιμοποιείται ως εξής

USAGE: perl update_iproclass.pl <dbuser> <dbpassword>

- dbuser: το όνομα του χρήστη της βάσης δεδομένων που έχει δικαιώματα να εισάγει δεδομένα
- dbpassword: ο κωδικός του χρήστη
- dbname: το όνομα της βάσης στην οποία θα καταχωρηθούν τα δεδομένα

2.3.3 Τα αρχεία XML και η επεξεργασία τους

Η Extensible Markup Language (XML) έχει τις ρίζες της στην διαχείριση εγγράφων και παράγεται από μια γλώσσα για δόμηση μεγάλων εγγράφων, που είναι γνωστή ως Standard Generalized Markup Language (SGML). Η XML μπορεί να αναπαραστήσει δεδομένα βάσεων δεδομένων, όπως επίσης πολλά άλλα είδη δομημένων δεδομένων που χρησιμοποιούνται σε επαγγελματικές εφαρμογές.

Με τη μεγάλη αποδοχή της XML ως αναπαράσταση δεδομένων και μορφή ανταλλαγής, χρησιμοποιούνται ευρέως εργαλεία προγραμμάτων για χειρισμό δεδομένων XML. Υπάρχουν δυο τυπικά μοντέλα χειρισμού της XML, όπου καθένα είναι διαθέσιμο για χρήση σε μια μεγάλη ποικιλία από δημοφιλείς γλώσσες προγραμματισμού..

Ένα από τα τυπικά API (Application Program Interface) για χειρισμό XML είναι το document object model (DOM), που χειρίζεται τα περιεχόμενα XML ως δένδρο, με κάθε στοιχείο να αντιπροσωπεύεται από ένα κόμβο, που ονομάζεται DOMNode. Τα προγράμματα μπορεί να έχουν πρόσβαση σε μέρη του εγγράφου ξεκινώντας από την ρίζα.

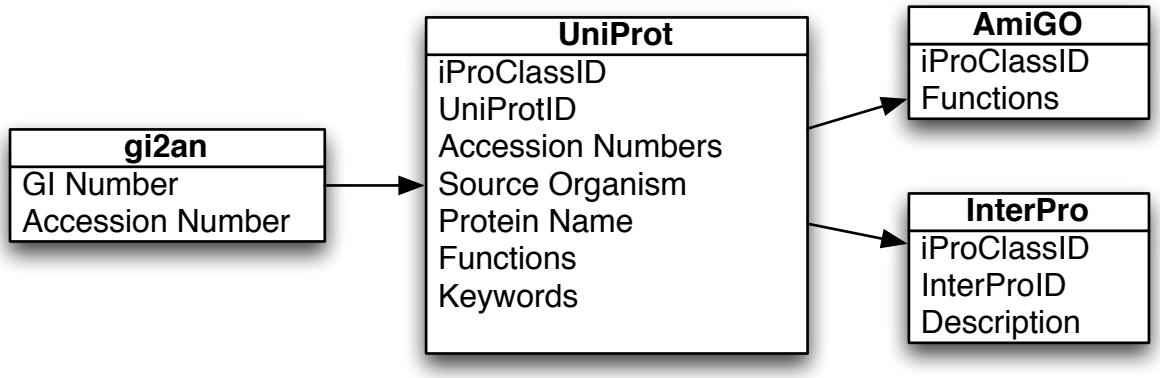
Η δεύτερη διασύνδεση προγραμματισμού (API) είναι το SAX (Simple API for XML). Το SAX είναι ένα μοντέλο συμβάντων, που έχει σχεδιαστεί να παρέχει μια κοινή διασύνδεση μεταξύ αναλυτών και εφαρμογών. Αυτό το API είναι δημιουργημένο με την έννοια των «χειριστών συμβάντων», που αποτελείται από συναρτήσεις καθορισμένες από το χρήστη που σχετίζονται με συμβάντα ανάλυσης. Τα συμβάντα ανάλυσης αντιστοιχούν στην αναγνώριση μερών ενός εγγράφου. Για παράδειγμα, δημιουργείται ένα συμβάν όταν βρίσκεται η ετικέτα αρχής για ένα στοιχείο κι ένα άλλο συμβάν όταν βρίσκεται η ετικέτα τέλους. Τα κομμάτια ενός εγγράφου πάντα συναντώνται με τη σειρά, από την αρχή ως το τέλος (streaming).

Από τις δυο λύσεις επιλέχθηκε η δεύτερη. Η αναπαράσταση των δεδομένων σε δέντρο DOM, αν και απλοποιεί πολύ την πρόσβαση στα δεδομένα και απαιτεί σημαντικά λιγότερο κώδικα, έχει μεγάλο υπολογιστικό κόστος και ιδιαίτερα σε μνήμη συστήματος. Αυτό συμβαίνει διότι για να έχει πρόσβαση ο χρήστης στα δεδομένα, πρέπει πρώτα να δημιουργηθεί ολόκληρο το δέντρο στη μνήμη του συστήματος. Όταν τα δεδομένα είναι της τάξης των μερικών GB (Giga Bytes) αυτό δεν είναι εφικτό. Συνεπώς επιλέχθηκε η δεύτερη λύση, ως πιο αποδοτική, αν κι αυξάνει την πολυπλοκότητα του κώδικα.

Για να επεξεργαστούμε το XML αρχείο χρησιμοποιήσαμε το XML::Parser::Expat module, μια βιβλιοθήκη για επεξεργασία XML αρχείων γραμμένη σε C. Εφόσον τα κομμάτια του εγγράφου συναντώνται με τη σειρά από την αρχή ως το τέλος και ο κώδικας μας βασίζεται σε «χειριστές συμβάντων», πρέπει να γνωρίζουμε την ακριβή δομή του XML αρχείου. Πρέπει να γνωρίζουμε τη σειρά με την οποία θα συναντήσουμε τα στοιχεία προς αποθήκευση, που θα είναι και η σειρά με την οποία θα αποθηκευτούν. Αυτές τις πληροφορίες μας τις δίνει το σχήμα XML (XMLSchema), ένα αρχείο που ορίζει τον τύπο του εγγράφου. Διατίθεται από το server του iProClass. Ακολουθεί ένα δείγμα του αρχείου

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<xs:schema  xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://pir.georgetown.edu/iproclass"
targetNamespace="http://pir.georgetown.edu/iproclass"
elementFormDefault="qualified">
      <xs:element name="iProClassDatabase">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element ref="iProClassEntry" maxOccurs="unbounded"/>
                  </xs:sequence>
                  <xs:attribute name="release_date" type="xs:string"/>
                  <xs:attribute name="version" type="xs:string"/>
            </xs:complexType>
      </xs:element>
      <xs:element name="iProClassEntry">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element ref="GENERAL_INFORMATION"/>
                        <xs:element ref="CROSS_REFERENCES" minOccurs="0"/>
                        <xs:element ref="FAMILY_CLASSIFICATION" minOccurs="0"/>
                        <xs:element ref="SEQUENCE"/>
                  </xs:sequence>
                  <xs:attribute name="Entry_ID" type="xs:ID" use="required"/>
            </xs:complexType>
      </xs:element>
      ...
</xs:schema>
```

Ο κώδικας που περιέχει τους event handlers (χειριστές συμβάντων), βρίσκεται στο βοηθητικό script **xmlparser.pl** (χρησιμοποιείται μέσα από το update_iproclass.pl). Στο τέλος, οι πληροφορίες αποθηκεύονται σε πίνακες στη βάση δεδομένων. Ακολουθεί διάγραμμα των πινάκων που χρησιμοποιούνται.



Όπως βλέπουμε, οι πληροφορίες που είναι διαθέσιμες είναι:

- **UniProt:** Protein Name, Source Organism, Function, Keywords
- **AmiGO**: Gene Ontology (ncbi)
- **InterPro:** Family Classification

2.4 Το web interface και η αλληλεπίδραση με το χρήστη

Το interface του προγράμματος είναι γραμμένο σε Perl με τη βοήθεια του module CGI.

Το Common Gateway Interface (CGI) είναι ένα σύστημα παραγωγής δυναμικών σελίδων html το οποίο αποτελεί υποσύνολο του Hypertext Transfer Protocol - http. Ο εξυπηρετητής WWW δέχεται ένα αίτημα WWW από τον πελάτη, παράγει δυναμικά την ιστοσελίδα ανάλογα με το αίτημα και την αποστέλλει στον client. Για τη δυναμική διαμόρφωση ιστοσελίδων χρησιμοποιούνται προγράμματα σε όλες τις δημοφιλείς γλώσσες προγραμματισμού, όπως C, C++, Java, όπως και Perl scripts.

2.4.1 Η αρχική σελίδα και τα δεδομένα εισόδου

Η πρώτη σελίδα που βλέπει ο χρήστης είναι η στατική *index.html*. Υπάρχει μια φόρμα μέσω της οποίας ο χρήστης μπορεί να εισάγει δεδομένα. Τα δεδομένα αυτά μπορεί να είναι τριών διαφορετικών μορφών.

1. Απλό αρχείο κειμένου με αναγνωριστικούς αριθμούς GI

```
Acc./id Description
gi|81862571 Acetyl-CoA carboxylase 1 (ACC-alpha) (Acetyl-CoA carboxylase 265)

gi|40556608 heat shock protein 1, beta [Mus musculus]

gi|6754254  heat shock protein 1, alpha [Mus musculus]
gi|74220416 unnamed protein product [Mus musculus]
gi|74220592 unnamed protein product [Mus musculus]
gi|13242237 heat shock protein 8 [Rattus norvegicus]
gi|42542422 Heat shock protein 8 [Mus musculus]
gi|74181586 unnamed protein product [Mus musculus]
gi|74181633 unnamed protein product [Mus musculus]
gi|74208631 unnamed protein product [Mus musculus]
gi|74143862 unnamed protein product [Mus musculus]
gi|74214176 unnamed protein product [Mus musculus]
gi|74141821 unnamed protein product [Mus musculus]
gi|50510319 mKIAA0002 protein [Mus musculus]
```

Το αρχείο κειμένου αρκεί να έχει μια πρωτεΐνη ανά γραμμή, με το GI id της. Ο αριθμός GI μπορεί να βρίσκεται οπουδήποτε στη γραμμή, αρκεί να έχει μπροστά το σχετικό αναγνωριστικό «gi|»

2. **Αρχείο Excel με αναγνωριστικούς αριθμούς GI**

C2

	A	B
1	**CE BirA.Field7**	**CE BirA.Field8**
2		
3	Acc./id	Description
4	gi|81862571	Acetyl-CoA carboxylase 1 (ACC-alpha) (Acetyl-CoA carboxylase 265) [Includes: Biotin carboxylase]
5		
6	gi|40556608	heat shock protein 1, beta [Mus musculus]
7		
8	gi|6754254	heat shock protein 1, alpha [Mus musculus]
9	gi|74220416	unnamed protein product [Mus musculus]
10	gi|74220592	unnamed protein product [Mus musculus]
11	gi|13242237	heat shock protein 8 [Rattus norvegicus]
12	gi|42542422	Heat shock protein 8 [Mus musculus]
13	gi|74181586	unnamed protein product [Mus musculus]
14	gi|74181633	unnamed protein product [Mus musculus]
15	gi|74208631	unnamed protein product [Mus musculus]
16	gi|74143862	unnamed protein product [Mus musculus]
17	gi|74214176	unnamed protein product [Mus musculus]
18	gi|74141821	unnamed protein product [Mus musculus]
19	gi|50510319	mKIAA0002 protein [Mus musculus]

Οι αριθμοί GI πρέπει να βρίσκονται στην πρώτη στήλη και τα δεδομένα να αρχίζουν από την 4η γραμμή.

3. Αρχείο Excel, σύγκριση Control και δείγματος.

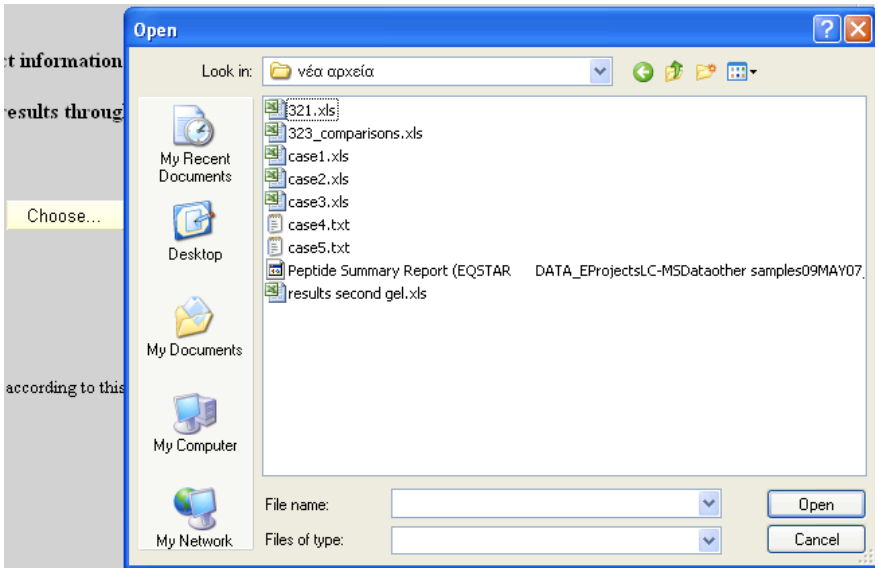
Στη μία στήλη (αριστερά) έχουμε τα κύτταρα ελέγχου (control), στη δεύτερη στήλη έχουμε τα κύτταρα που μας ενδιαφέρουν. Οι πρωτεΐνες που είναι μοναδικές στη δεύτερη στήλη είναι οι

	A	B	C	D	E	F	G	H
1	1.Field5	1.Field6	1.Field7	1.Field8	2.Field5	2.Field6	2.Field7	2.Field8
2								
3	Hit #	Score	Acc./id	Description	Hit #	Score	Acc./id	Description
4	1	4493	gi|81862571	Acetyl-CoA carboxylase 1 (ACC-alpha) (Acetyl-CoA	1	4344	gi|81862571	Acetyl-CoA carboxylase 1 (ACC-alpha) (Acetyl-CoA carboxylase 265) [Inc
5	100	77	gi|73921193	Myosin-9 (Myosin heavy chain, nonmuscle IIa) (Nonr	2	1335	gi|73921193	Myosin-9 (Myosin heavy chain, nonmuscle IIa) (Nonmuscle myosin heavy c
6	100	77	gi|11432644	myosin, heavy polypeptide 9, non-muscle isoform 1	2	1335	gi|11432644	myosin, heavy polypeptide 9, non-muscle isoform 1 [Mus musculus]
7	100	77	gi|74180977	unnamed protein product [Mus musculus]	2	1332	gi|74180977	unnamed protein product [Mus musculus]
8	4	796	gi|40556608	heat shock protein 1, beta [Mus musculus]	3	1332	gi|40556608	heat shock protein 1, beta [Mus musculus]
9	3	797	gi|5174735	tubulin, beta, 2 [Homo sapiens]	4	1244	gi|5174735	tubulin, beta, 2 [Homo sapiens]
10	3	797	gi|13542680	Tubulin, beta 2c [Mus musculus]	4	1244	gi|13542680	Tubulin, beta 2c [Mus musculus]
11	2	866	gi|7106439	tubulin, beta 5 [Mus musculus]	5	1109	gi|7106439	tubulin, beta 5 [Mus musculus]
12	2	866	gi|12846758	unnamed protein product [Mus musculus]	5	1109	gi|12846758	unnamed protein product [Mus musculus]
13	2	866	gi|74141821	unnamed protein product [Mus musculus]	5	1109	gi|74141821	unnamed protein product [Mus musculus]
14	2	866	gi|74204140	unnamed protein product [Mus musculus]	5	1109	gi|74204140	unnamed protein product [Mus musculus]
15	2	866	gi|74223737	unnamed protein product [Mus musculus]	5	1109	gi|74223737	unnamed protein product [Mus musculus]
16	13	449	gi|6754254	heat shock protein 1, alpha [Mus musculus]	6	1064	gi|6754254	heat shock protein 1, alpha [Mus musculus]
17	13	449	gi|74147335	unnamed protein product [Mus musculus]	6	1064	gi|74147335	unnamed protein product [Mus musculus]

πιθανές αλληλεπιδρούσες πρωτεΐνες με το STAT5. Για το λόγο αυτό μας ενδιαφέρει να γίνει αυτόματη συλλογή πληροφοριών για τις συγκεκριμένες μοναδικές πρωτεΐνες της δεύτερης στήλης.

Όσο πιο υψηλό είναι το score, τόσο πιο μεγάλη είναι η αξιοπιστία του ευρήματος και η πιθανότητα να είναι πραγματική η πρωτεΐνη. Πρωτεΐνες με score μικρότερο του 40, συνήθως δεν αναλύονται περαιτέρω πειραματικώς στο εργαστήριο.

Τα δεδομένα πρέπει να αρχίζουν από την 4η γραμμή και οι στήλες να έχουν τη σειρά που φαίνεται στην παραπάνω εικόνα (Hits, Score, ID, Description, Hits, Score, ID, Description). Το κάθε αρχείο μπορεί να περιέχει περισσότερα από ένα πειράματα, σε διαφορετικά excel worksheets, τα αποτελέσματα των οποίων θα παρουσιαστούν ξεχωριστά μετά την επεξεργασία των δεδομένων. Ο χρήστης πατώντας «Choose», καλείται να επιλέξει το αρχείο με τα δεδομένα προς επεξεργασία. Καθορίζει το είδος το αρχείου από το drop-down μενού, επιλέγει «Submit» κι αναμένει τα αποτελέσματα.



2.4.2 Η επεξεργασία των δεδομένων - upload.cgi

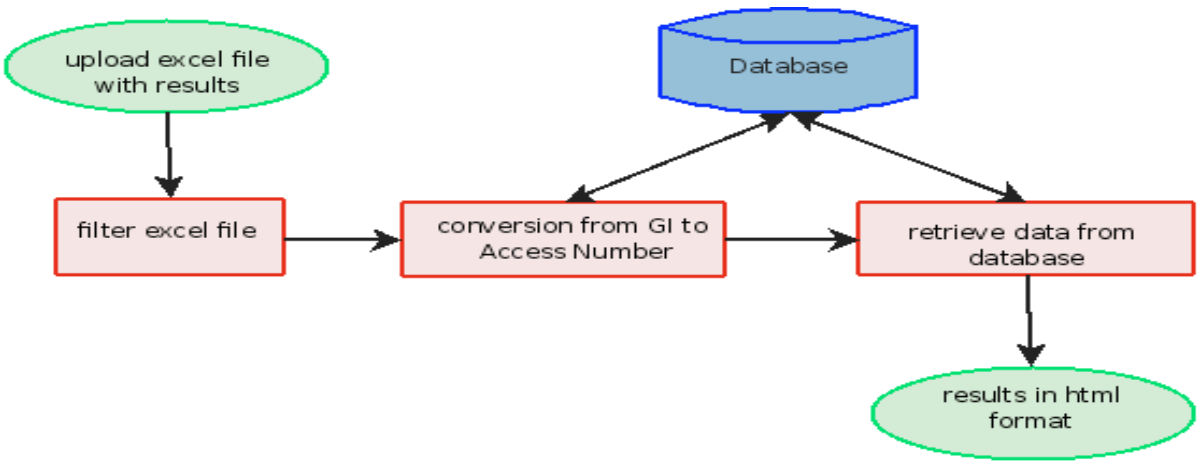
Το κουμπί «Submit» καλεί το πρόγραμμα *upload.cgi*. Περνά σε αυτό το όνομα του αρχείου καθώς και το είδος του, δεδομένα που έχει καταχωρήσει ο χρήστης.

U P L O A D . C G I

Τα modules που χρησιμοποιούνται είναι τα εξής

- CGI, το οποίο μας παρέχει τις συναρτήσεις και μεταβλητές για το cgi script μας
- Win32::OLE, module που μας επιτρέπει να χειριστούμε εφαρμογές της Microsoft, σε αυτή τη περίπτωση το Microsoft Excel
- DBI, το module που μας επιτρέπει να επικοινωνήσουμε με τη βάση δεδομένων που έχουμε στήσει
- Archive::Zip, module που μας επιτρέπει να συμπιέσουμε ένα αριθμό αρχείων. Χρησιμοποιείται για να συμπιεστεί και ομαδοποιηθεί το σύνολο των αποτελεσμάτων, ώστε να τα αποθηκεύσει ο χρήστης.

Ακολουθεί διάγραμμα που περιγράφει τη λειτουργία του upload.cgi και στη συνέχεια αναλυτική εξήγηση του κάθε βήματος.



1. FILTER EXCEL FILE

Αφού ο χρήστης έχει πατήσει το κουμπί «Submit», το script upload.cgi αναλαμβάνει να εξάγει από αυτό τις απαραίτητες πληροφορίες. Η βασική πληροφορία που χρειαζόμαστε είναι οι αριθμοί GI, οι οποίοι αρκούν για να εντοπιστούν οι πρωτεΐνες στις βάσεις δεδομένων. Αυτή την πληροφορία την περιέχουν όλα τα αρχεία που δέχεται η εφαρμογή.

Το αρχείο που περιέχει τη σύγκριση των κυττάρων ελέγχου (control) με το δείγμα, περιέχει και τις πληροφορίες «hits» και «score», τα οποία επίσης αποθηκεύονται ώστε να έχουμε τα δυνατότητα στη συνέχεια να ταξινομήσουμε τα αποτελέσματα με βάση τα παραπάνω χαρακτηριστικά.

Η ταξινόμηση είναι σημαντική διότι όσο πιο υψηλό είναι το score, τόσο πιο μεγάλη είναι η αξιοπιστία του ευρήματος και η πιθανότητα να είναι πραγματική η πρωτεΐνη.

Διαφοροποιήσεις του αλγορίθμου ανάλογα με τον τύπο των δεδομένων

Στην περίπτωση που τα δεδομένα εισόδου είναι **Τύπου 1** (αρχείο απλού κειμένου), η εξαγωγή των δεδομένων είναι απλή. Στην περίπτωση δεδομένων **Τύπου 2** (αρχείο Excel με αναγνωριστικούς αριθμούς GI), πρέπει να χρησιμοποιηθεί το Win32::OLE module της Perl. Στην περίπτωση δεδομένων **Τύπου 3** (αρχείο Excel με σύγκριση Control και δείγματος), απαιτείται ένα ακόμα βήμα, η σύγκριση των δυο στηλών. Όπως αναφέρθηκε προηγουμένως, μας ενδιαφέρουν οι πρωτεΐνες που υπάρχουν μόνο στη δεξιά στήλη και γι' αυτές και μόνο θα αναζητήσουμε βιβλιογραφικές πληροφορίες.

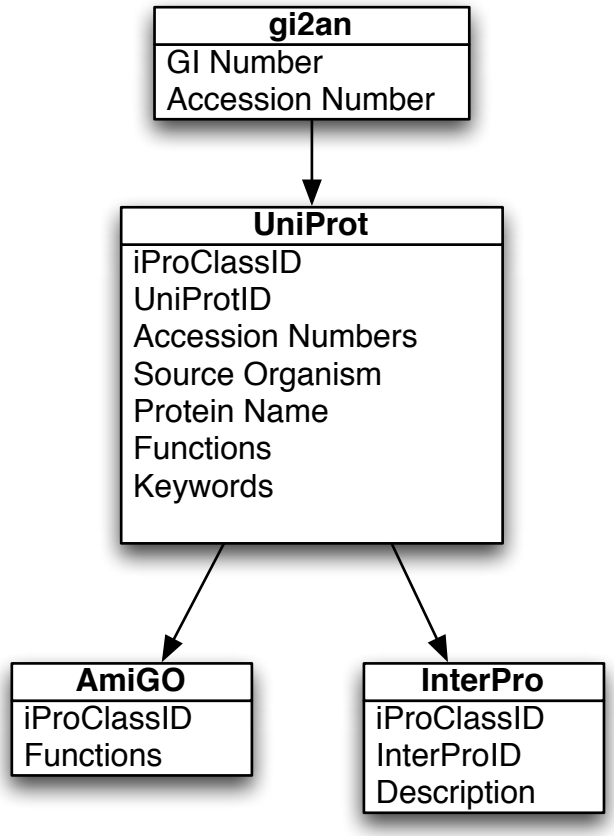
2. CONVERSION FROM GI NUMBER TO ACCESSION NUMBER

Από το προηγούμενο βήμα έχουμε στη διάθεσή μας το σύνολο των GI αναγνωριστικών αριθμών των πρωτεϊνών που μας ενδιαφέρουν. Ο αναγνωριστικός αριθμός GI χρησιμοποιείται στη βάση δεδομένων του NCBI, όχι όμως και στις άλλες βάσεις που μας ενδιαφέρουν. Σε αυτό το βήμα, τον μετατρέπουμε με τη βοήθεια του πίνακα iproclass.tb σε Accession Number που μας επιτρέπει την επικοινωνία με τις υπόλοιπες βάσεις. Για κάθε αριθμό GI, γίνεται ένα query στη βάση δεδομένων, στον πίνακα gi2an. Η απάντηση αποθηκεύεται και τα ερωτήματα (queries) επαναλαμβάνονται για όλες τις πρωτεΐνες.

3. RETRIEVE DATA FROM DATABASE

Με βάση τα accession numbers, γίνονται νέα ερωτήματα στη βάση δεδομένων κι αποθηκεύονται χωριστά για την κάθε πρωτεΐνη. Όταν αυτό το βήμα ολοκληρωθεί, οι πληροφορίες που έχουν καταχωρηθεί για την κάθε πρωτεΐνη είναι οι εξής:

- GI Number
- HIT*
- SCORE*
- Accession Number
- Uniprot ID
- Source OrganismProtein Name
- iProClass ID
- Keywords
- Functions (ontology)
- InterPro ID
- InterPro Description

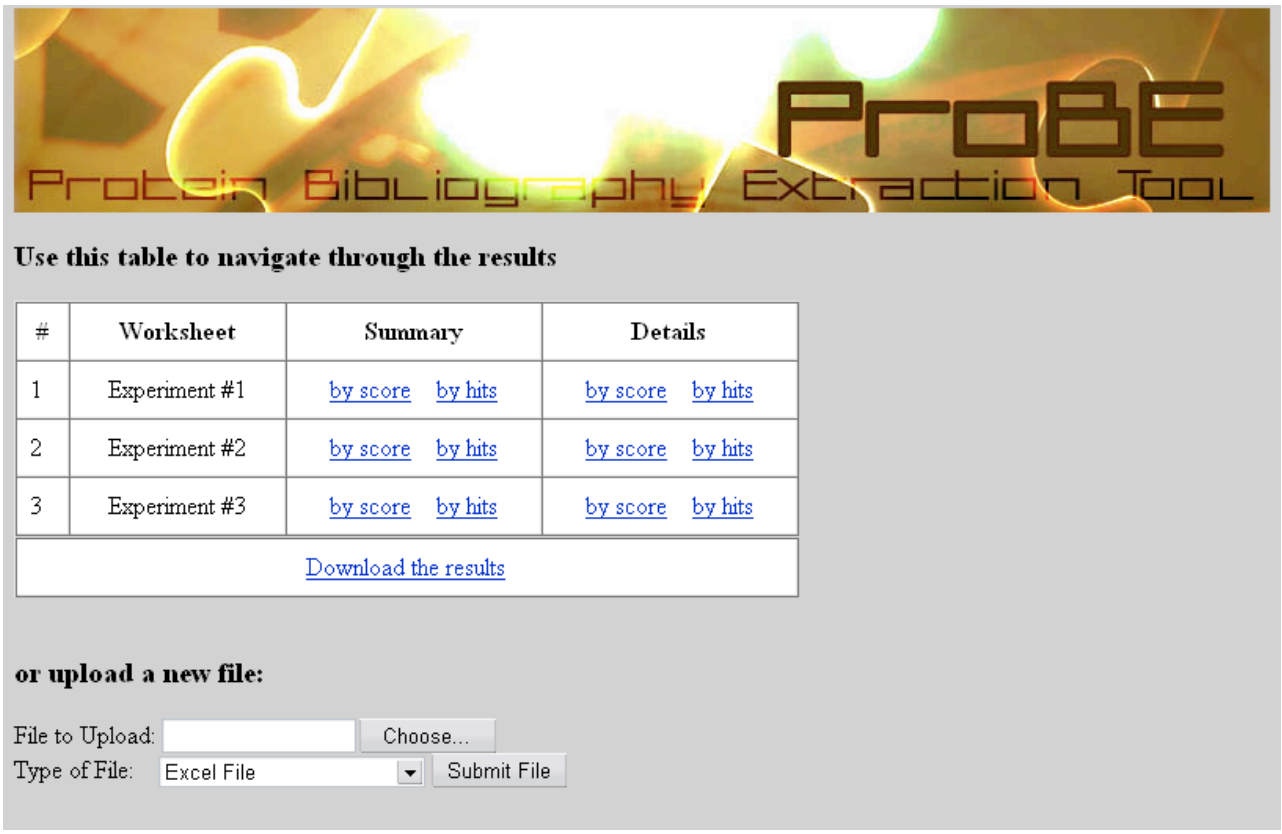


*οι πληροφορίες αυτές περιέχονται μόνο στο συγκριτικό αρχείο excel

Το τελευταίο βήμα είναι η εκτύπωση των αποτελεσμάτων.

2.4.3 Εκτύπωση των αποτελεσμάτων

Σε αυτό το βήμα, τα αποτελέσματά μας είναι έτοιμα προς εκτύπωση. Τυπώνεται ο παρακάτω πίνακας.



Από εδώ ο χρήστης μπορεί να διαλέξει, ποιο πείραμα (worksheet) θέλει να εξετάσει πρώτα. Μπορεί να δει είτε περιληπτικά τις πρωτεΐνες ταξινομημένες ανά hits/score (summary), είτε όλες τις διαθέσιμες πληροφορίες (details).

Ακόμα, μπορεί να αποθηκεύσει τα παραπάνω αποτελέσματα σε μορφή .**zip**, καθώς και να εισάγει νέα δεδομένα προς ανάλυση. Η εκτύπωση των αποτελεσμάτων γίνεται με τη χρήση του module CGI.

Ακολουθεί ένα παράδειγμα παρουσίασης συνοπτικών αποτελεσμάτων. Αυτή η επιλογή υπάρχει μόνο στην περίπτωση συγκριτικού αρχείου excel, όπου και παρουσιάζονται μόνο οι πρωτεΐνες που μας ενδιαφέρουν, ταξινομημένες ανά hits ή score, ανάλογα με την επιλογή του χρήστη.

Συνοπτικά αποτελέσματα

#	Worksheet	Summary	Details
1	First Experiment	by score by hits	by score by hits
2	Second Experiment	by score by hits	by score by hits
3	Third Experiment	by score by hits	by score by hits
Download the results			

or upload a new file:

File to Upload: Choose...
Type of File: Excel File Submit File

This is worksheet #1, called "*First Experiment* "

The results sorted by score are:

Score	Hits	Protein
4344	1	Acetyl-CoA carboxylase 1 (ACC-alpha) (Acetyl-CoA carboxylase 265) [Includes: Biotin carboxylase]
1335	2	Myosin-9 (Myosin heavy chain, nonmuscle IIa) (Nonmuscle myosin heavy chain IIa) (NMMHC II-a) (NMMHC
1335	2	myosin, heavy polypeptide 9, non-muscle isoform 1 [Mus musculus]
1332	3	heat shock protein 1, beta [Mus musculus]
1332	2	unnamed protein product [Mus musculus]
1244	4	tubulin, beta, 2 [Homo sapiens]
1244	4	Tubulin, beta 2c [Mus musculus]

Παρατηρούμε πως ο πίνακας που περιέχει το σύνολο των αποτελεσμάτων βρίσκεται στην αρχή της σελίδας, ώστε να μπορεί ο χρήστης να μεταφέρεται ανάμεσα στα διαφορετικά πειράματα και τους διαφορετικούς τρόπους παρουσίασης ταξινόμησης. Ακόμα, σε κάθε σελίδα παρουσίασης αποτελεσμάτων, υπάρχει η επιλογή για νέα αναζήτηση στη βάση δεδομένων.

Ακολουθεί ένα παράδειγμα λεπτομερούς παρουσίασης των αποτελεσμάτων.

e this table to navigate through the results

	Worksheet	Summary	Details
	First Experiment	by score by hits	by score by hits
	Second Experiment	by score by hits	by score by hits
	Third Experiment	by score by hits	by score by hits

Download the results

upload a new file:

e to Upload: Choose...
pe of File: Excel File Submit File

is is worksheet #1, called "*First Experiment* "

detailed view of the results sorted by score follows

il81862571 has 1 hits and a 4344 score

rotein Name and rganism	*Name:* Acetyl-CoA carboxylase 1 (EC 6.4.1.2) (ACC-alpha) (Acetyl-CoA carboxylase 265) [Includes: Biotin carboxylase (EC 6.3.4.14)] *Organism:* Mus musculus (Mouse)
eywords	*Keywords:* atp-binding , biotin , direct protein sequencing , fatty acid biosynthesis , ligase , lipid synthesis , manganese , metal-binding , multifunctional enzyme , nucleotide-binding , phosphorylation
unction	*Function:* Catalyzes the rate-limiting reaction in the biogenesis of long-chain fatty acids. Carries out three functions: biotin carboxyl carrier protein, biotin carboxylase and carboxyltransferase (By similarity).
ntology	*Ontology:* acetyl-CoA carboxylase activity , catalytic activity , ATP binding , nucleotide binding , manganese ion binding , biotin carboxylase activity , ligase activity , biotin binding , metal ion binding , lipid biosynthetic process , metabolic process , fatty acid biosynthetic process , acetyl-CoA carboxylase activity , catalytic activity , ATP binding , nucleotide binding , manganese ion binding , biotin carboxylase activity , ligase activity , biotin binding , metal ion binding , lipid biosynthetic process , metabolic process , fatty acid biosynthetic process
amily Classification	*InterPro :* Acetyl-CoA carboxylase, central region , ATP-grasp fold , ATP-grasp fold, subdomain 2 , Biotin carboxylation region , Biotin-binding site , Biotin carboxylase, C-terminal , Biotin/lipoyl attachment , Carboxyl transferase , Acetyl-coenzyme A carboxyltransferase, C-terminal , Acetyl-coenzyme A carboxyltransferase, N-terminal , Carbamoyl-phosphate synthetase large chain, N-terminal , Carbamoyl-phosphate synthase L chain, ATP-binding , Single hybrid motif , Pre-ATP-grasp fold , Rudiment single hybrid motif
xternal Links	*NCBI :* 81862571 *UniProt :* COA1_MOUSE *MGI :* Q5SWU9 *Ensembl :* Q5SWU9

Εκτός από τις πληροφορίες από τις βάσεις δεδομένων UniProt, AmiGO, iProClass, παρατίθονται και σύνδεσμοι προς τις ιστοσελίδες άλλων βάσεων, για διασταύρωση πηγών ή για εύρεση περισσότερων πληροφοριών.

Οι βάσεις είναι οι

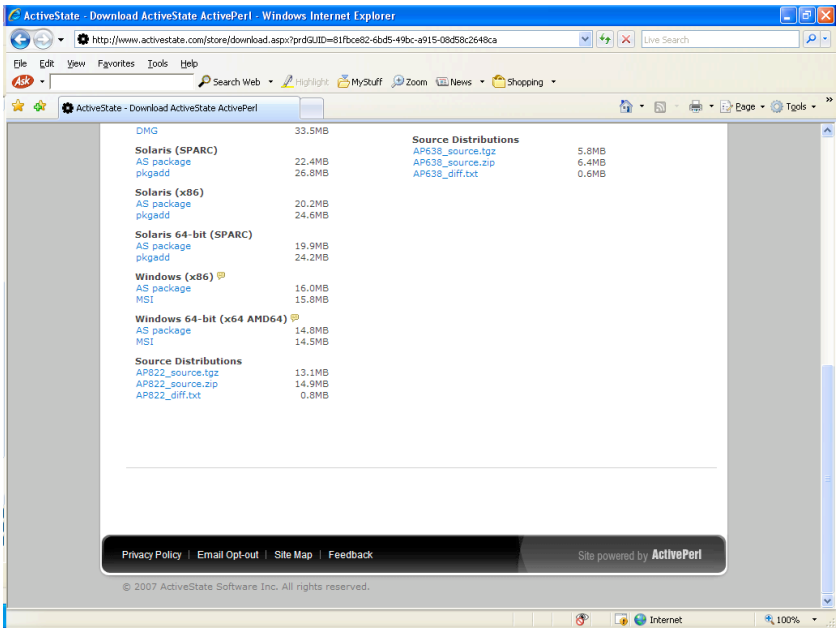
- NCBI, National Center for Biotechnology Information
- UniProt, Universal Protein Resource
- iProClass, PIR - Protein Information Resource
- MGI, Mouse Genome Informatics
- Ensembl, Ensemble Genome Browser

3.ΕΓΚΑΤΑΣΤΑΣΗ

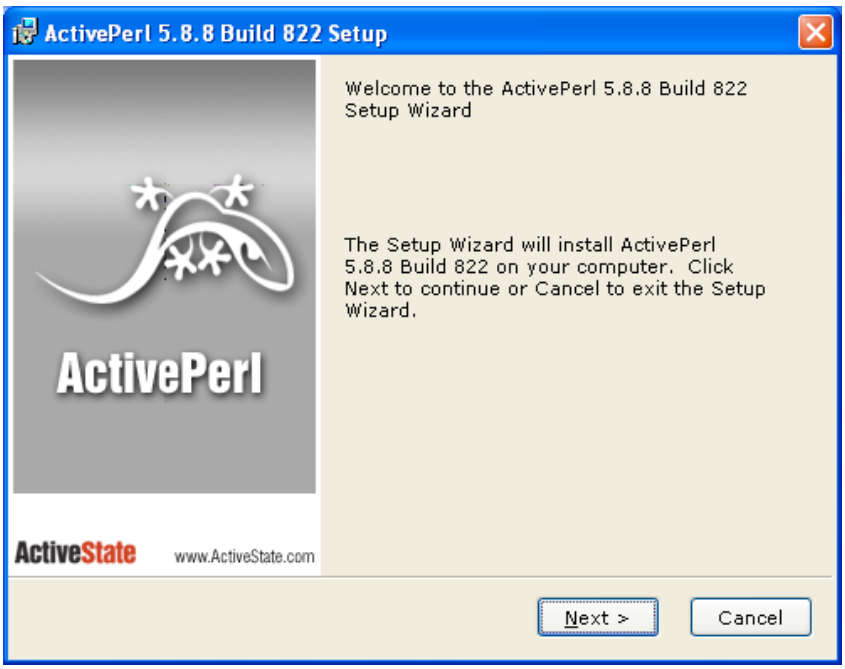
3.1 Εγκατάσταση της Perl

Προμηθευόμαστε τον installer της Perl από τη διεύθυνση

http://www.activestate.com/store/activeperl/download/

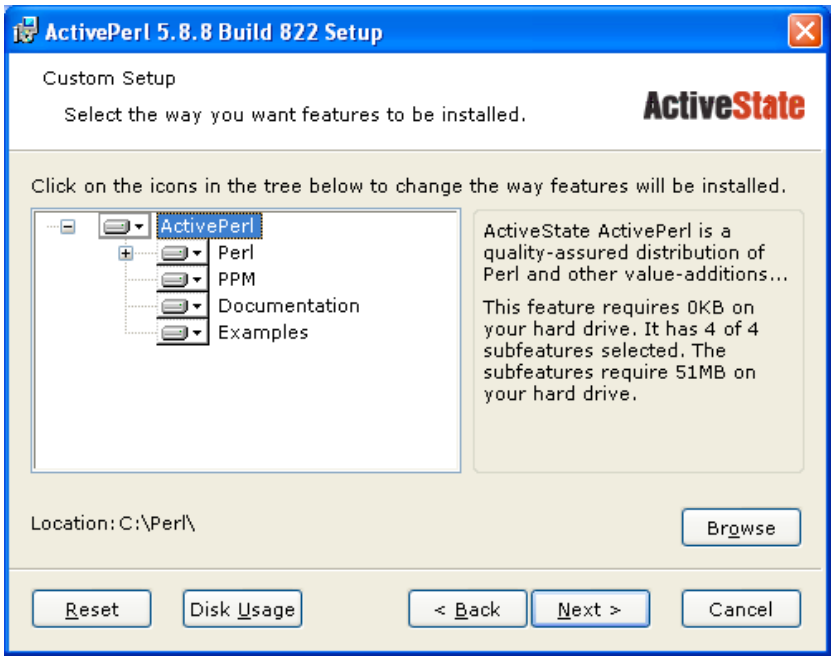


Κατεβάζουμε το πακέτο για windows και κάνουμε διπλό κλικ.



Πατάμε Next

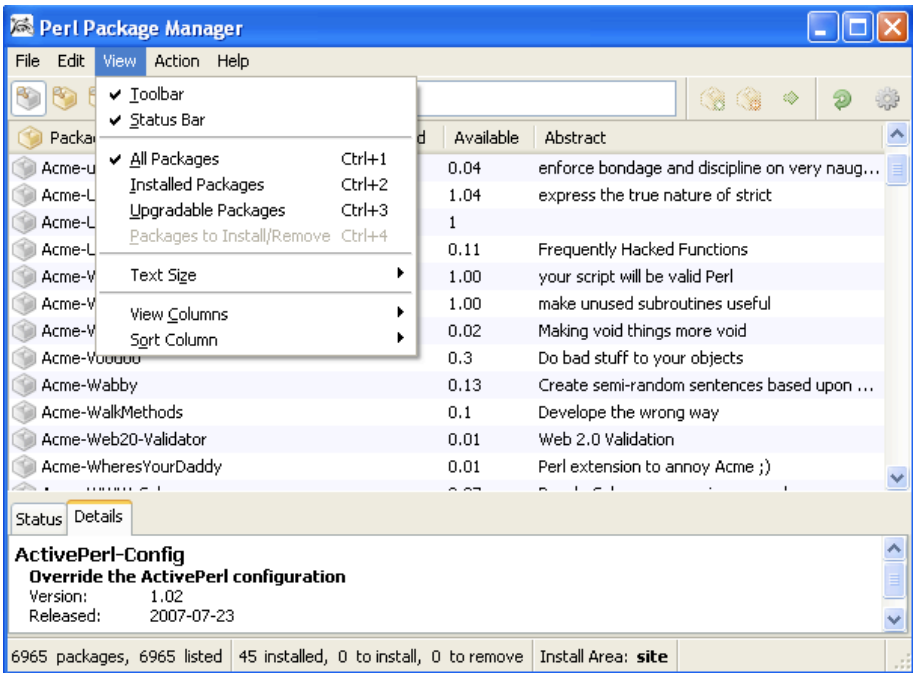
Αφήνουμε όλες τις προεπιλεγμένες ρυθμίσεις και συνεχίζουμε σε εγκατάσταση με το Next.



Όταν αυτό το βήμα ολοκληρωθεί, θα έχουμε μια λειτουργική εγκατάσταση της Perl. Μένει να εγκαταστήσουμε τα ξεχωριστά modules που χρησιμοποιούμε στον κώδικά μας.

Πηγαίνουμε στο μενού «Έναρξη» των Windows κι επιλέγουμε ActivePerl και Perl Package Manager, όπως φαίνεται στο διπλανό σχήμα.

Εμφανίζεται ο Package Manager της Perl και φροντίζουμε να είναι επιλεγμένη η προβολή όλων των πακέτων, όπως φαίνεται στη διπλανή εικόνα. Με δεξί κλικ σε οποιοδήποτε πακέτο και "install package", εγκαθίσταται το ζητούμενο module. Φροντίζουμε να είναι εγκατεστημένα τα



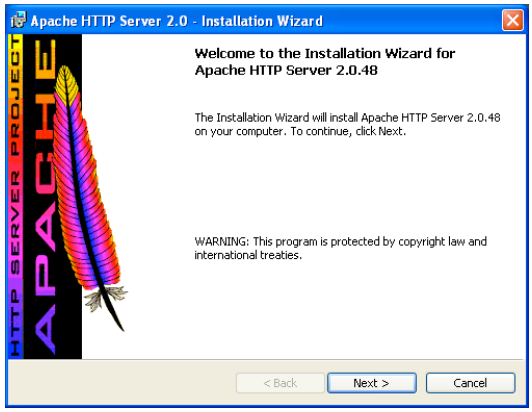
- DBI
- Archive-Zip
- XML-Parser

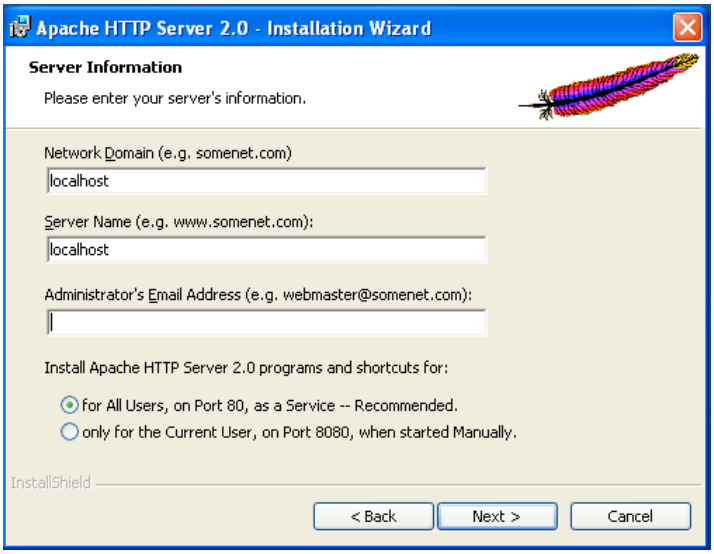
3.2 Εγκατάσταση του Apache Web Server

Προμηθευόμαστε τον installer του Apache από τη διεύθυνση

http://httpd.apache.org/download.cgi

Με διπλό κλικ ξεκινά η εγκατάσταση





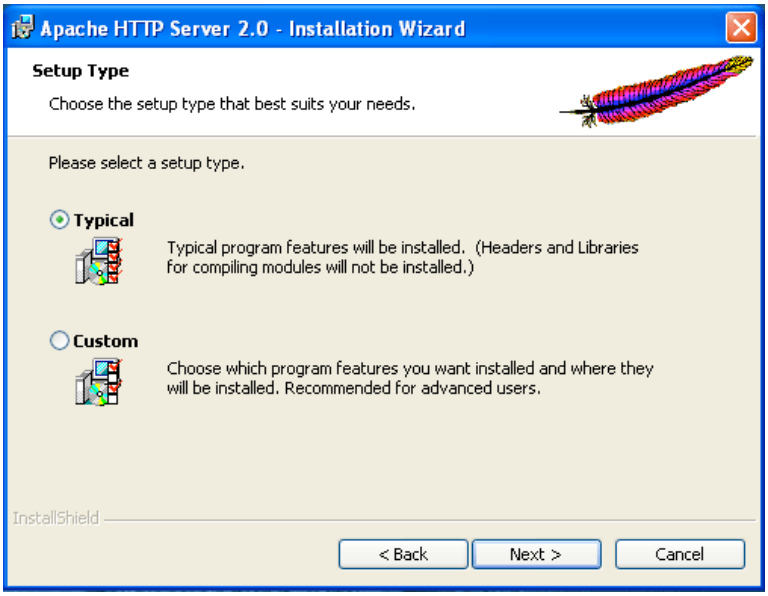
Θέτουμε:

Network Domain: localhost

Server Name: localhost

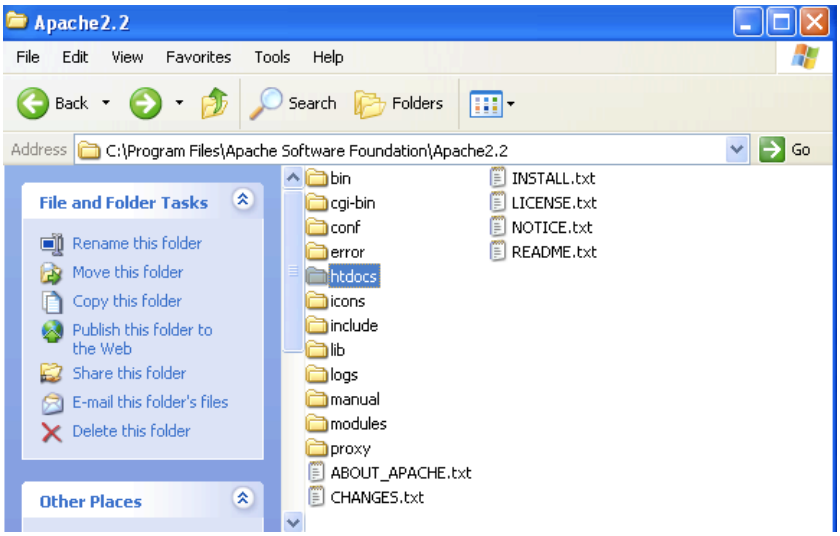
Μπορούμε να αφήσουμε κενό το "Administrator's Email Address", ενώ φροντίζουμε να είναι επιλεγμένο το "for All users, on Port 80, as a service"

Στην επόμενη οθόνη, επιλέγουμε "Typical". Δεν χρειάζεται να αλλάξουμε οτιδήποτε άλλο ως το τέλος της εγκατάστασης και επιλέγουμε "Next" μέχρι το τέλος της.



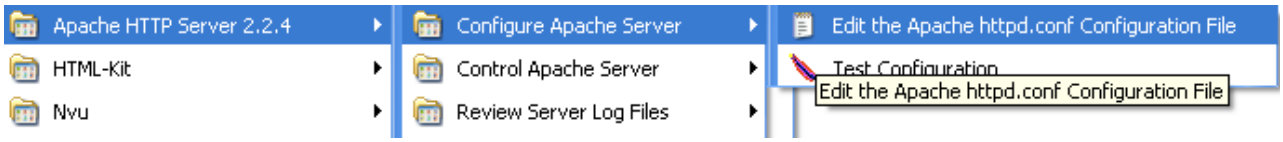
Σε αυτό το σημείο, ο Apache Web Server είναι εγκατεστημένος.

Αντιγράφουμε στο φάκελο *C:\Program Files\Apache Software Foundation\Apache2.2\htdocs* τα περιεχόμενα του φακέλου htdocs που περιέχονται στο φάκελο htdocs του συνοδευτικού CD εγκατάστασης της εφαρμογής μας. Τα περιεχόμενα περιλαμβάνουν τα αρχεία



- index.html
- upload.cgi
- protein-banner-illustration.jpg
- settings.png

Στη συνέχεια, πηγαίνουμε στο μενού «Έναρξη» των Windows κι επιλέγουμε *"Edit the Apache httpd.conf Configuration File"*.



Αναζητούμε το section "*<Directory "C:/Program Files/Apache Software Foundation/Apache2.2/htdocs">*", αν θέλουμε χρησιμοποιώντας τη search λειτουργία για ευκολία.

Εκεί υπάρχει ένα «σχολιασμένο» κομμάτι κειμένου

```
#  AllowOverride FileInfo AuthConfig Limit
#  Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
#  <Limit GET POST OPTIONS PROPFIND>
#     Order allow,deny
#     Allow from all
#  </Limit>
#  <LimitExcept GET POST OPTIONS PROPFIND>
#     Order deny,allow
#     Deny from all
#  </LimitExcept>
#</Directory>
```

Αφαιρούμε τους χαρακτήρες "#" από κάθε γραμμή, κι αλλάζουμε τη γραμμή "*Options*" με

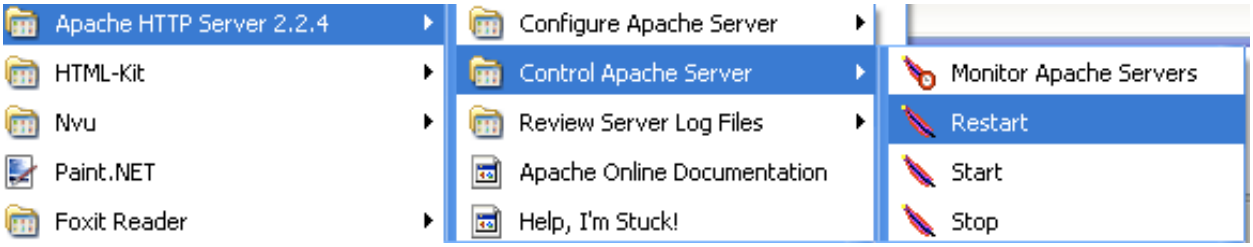
```
Options MultiViews Indexes SymLinksIfOwnerMatch Includes ExecCGI
```

Με τη βοήθεια της επιλογής Find (του notepad ή οποιουδήποτε άλλου προγράμματος χρησιμοποιούμε), βρίσκουμε τη γραμμή που περιέχει τη λέξη *"AddHandler"* κι αφαιρούμε το χαρακτήρα "#".

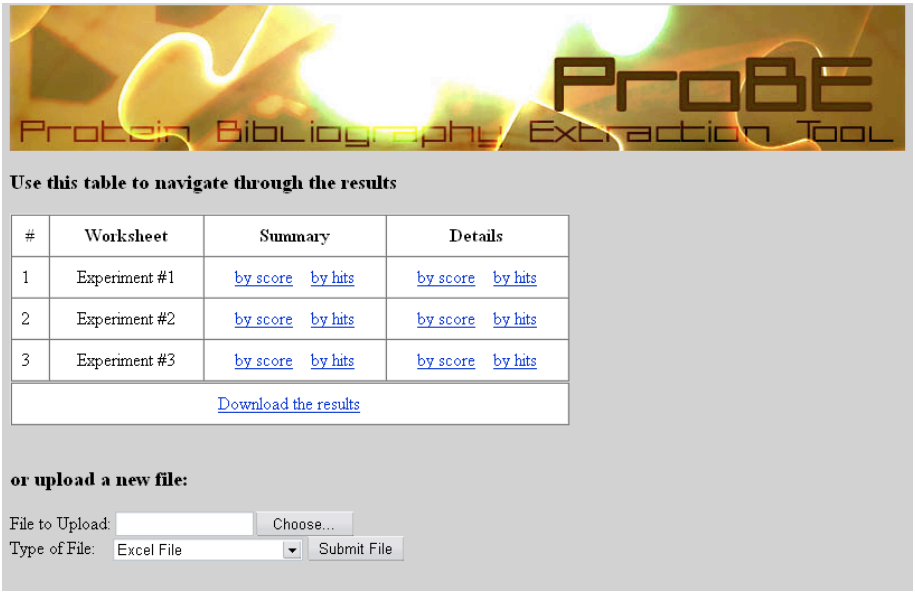
Η γραμμή αυτή θα πρέπει τώρα να είναι

```
AddHandler cgi-script .cgi .pl
```

Κλείνουμε το αρχείο και επανεκκινούμε τον Apache Web Server επιλέγοντας "Restart"



Πλέον έχουμε πρόσβαση στον web server μας, από οποιονδήποτε web browser, χρησιμοποιώντας ως διεύθυνση την *http://localhost*



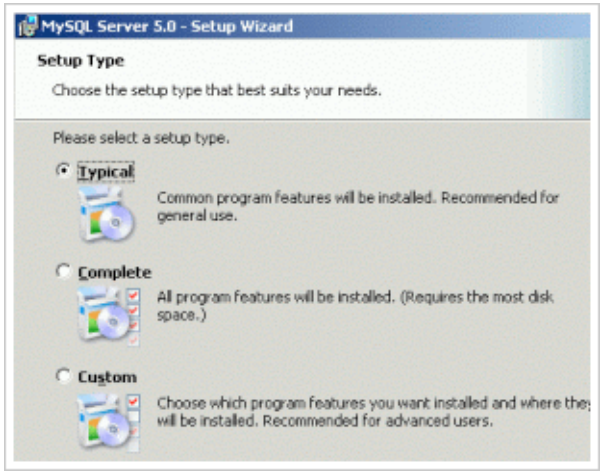
Εκεί θα δούμε την αρχική σελίδα της εφαρμογής μας. Δεν μπορούμε να τη χρησιμοποιήσουμε ακόμα, διότι δεν έχουμε εγκαταστήσει ακόμα τη βάση δεδομένων, το οποίο και θα κάνουμε στο επόμενο βήμα.

3.3 Εγκατάσταση της MySQL

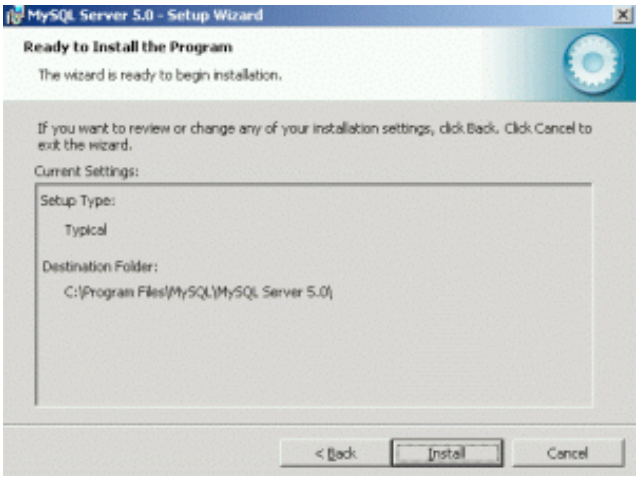
Προμηθευόμαστε το εκτελέσιμο της εγκατάστασης της MySQL από τη διεύθυνση

http://dev.mysql.com/downloads/mysql/5.0.html#win32

Ξεκινώντας την εγκατάσταση, επιλέγουμε αρχικά “Next” και στη συνέχεια “Typical” τρόπο εγκατάστασης.

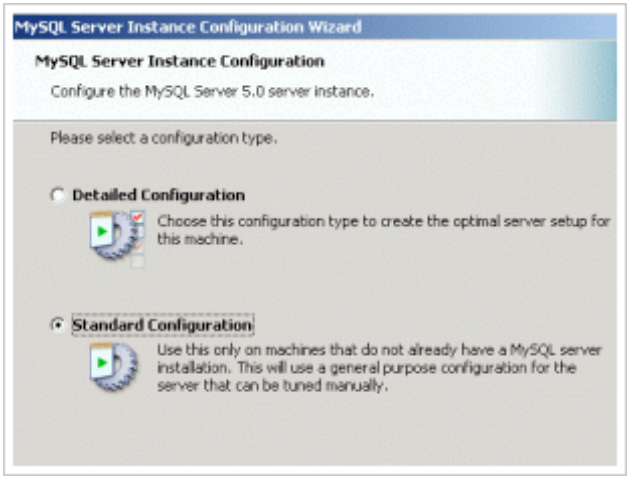


Στη συνέχεια επιλέγουμε “Install” και “Configure the MySQL Server now”.

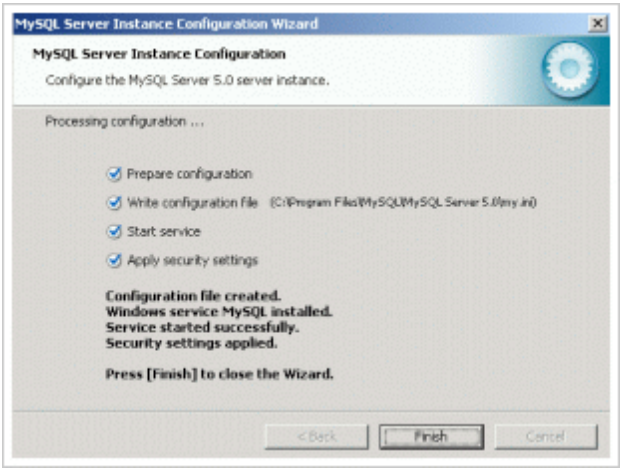


Επιλέγουμε “Standard Configuration” και στην επόμενη οθόνη βάζουμε τικ στα

- “Install As Windows Service”
- “Launch the MySQL Server automatically”
- “Include Bin Directory in Windows PATH”

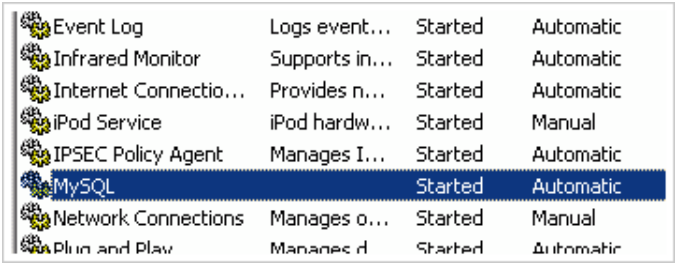


Στη συνέχεια επιλέγουμε “Modify Security Settings” και βάζουμε ως νέο root password τη λέξη “probe” χωρίς τα εισαγωγικά.

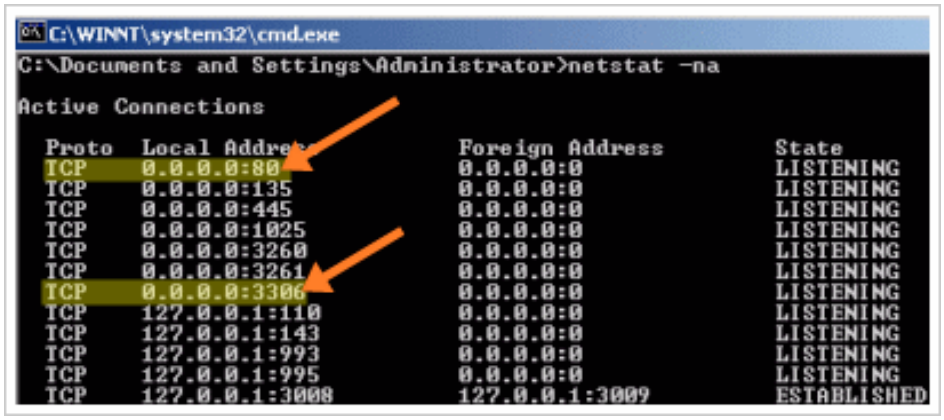


Η τελική οθόνη της εγκατάστασης μας πληροφορεί για σφάλματα που τυχόν προέκυψαν. Αν υπάρχει κάποιο σφάλμα, απενεργοποιούμε το software firewall και antivirus και επαναλαμβάνουμε τη διαδικασία της εγκατάστασης.

Για να επιβεβαιώσουμε πως η εγκατάσταση έγινε σωστά, πήγαινουμε στα "Administrative Tools" των Windows (Start -> Programs -> Administrative Tools -> Services). Θα πρέπει το service της MySQL να έχει ξεκινήσει ("Started"). Το ίδιο θα πρέπει να ισχύει και για το service του Apache Web Server.



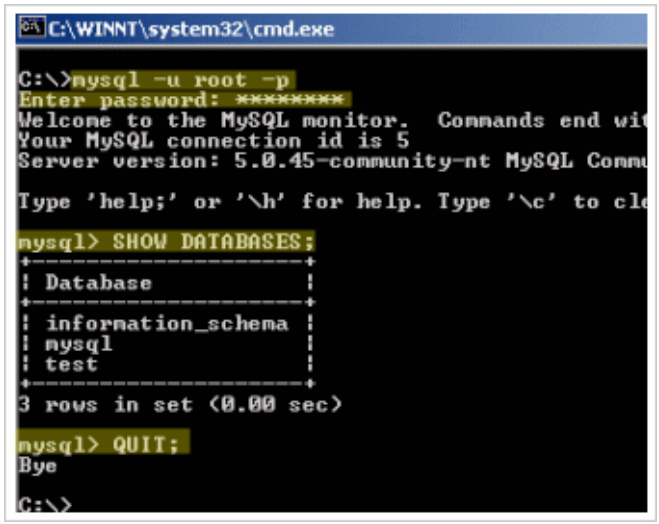
Ακόμα, μπορούμε να ανοίξουμε το Command Prompt των Windows (Start->Run -> cmd) και να πληκτρολογήσουμε την εντολή *"netstat -na"*. Επαληθεύουμε πως οι θύρες 3306 και 80 είναι ανοιχτές, όπως στην παρακάτω εικόνα.



Στη συνέχεια εισάγουμε την εντολή *"mysql -u -root -p"*, για να συνδεθούμε στη βάση δεδομένων που μόλις εγκαταστήσαμε. Όταν μας ζητηθεί password, εισάγουμε τη λέξη "probe", χωρίς τα εισαγωγικά. Εφόσον συνδεθούμε στη βάση δεδομένων, εισάγουμε την εντολή

"SHOW DATABASES;"

Εφόσον το αποτέλεσμα είναι το ίδιο με της διπλανής εικόνας, η βάση δεδομένων λειτουργεί και μπορούμε να προχωρήσουμε στο επόμενο βήμα, που είναι η εγκατάσταση των δεδομένων μας.



3.4 Εισαγωγή των δεδομένων στη βάση δεδομένων

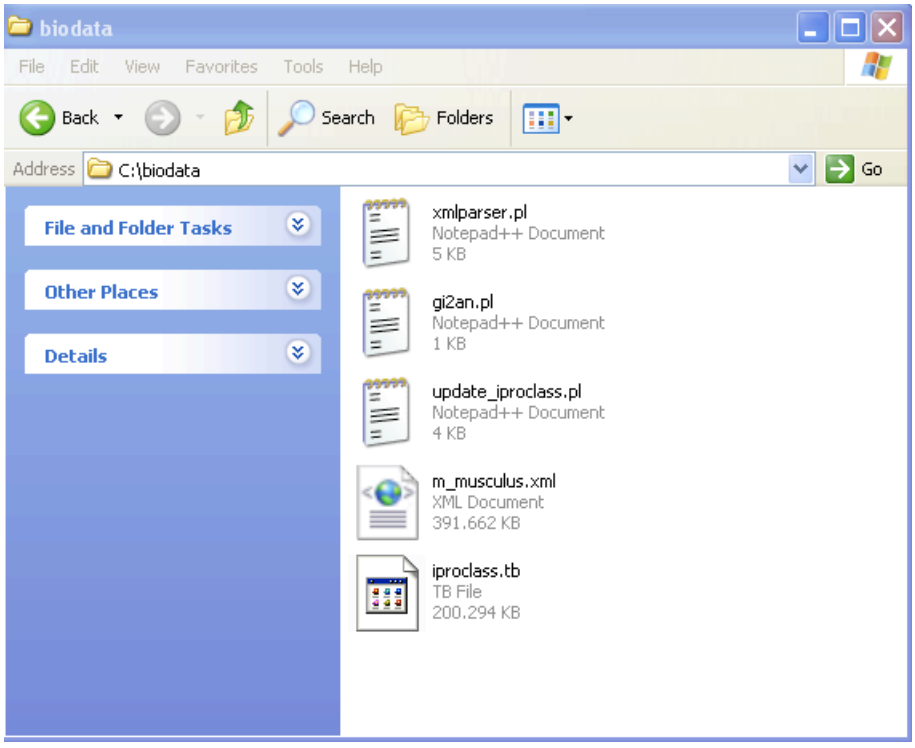
Συνεχίζουμε στη γραμμή εντολών και εισάγουμε τις εντολές

1. cd /
2. mkdir biodata
3. cd biodata

Σε αυτόν το φάκελο τοποθετούμε

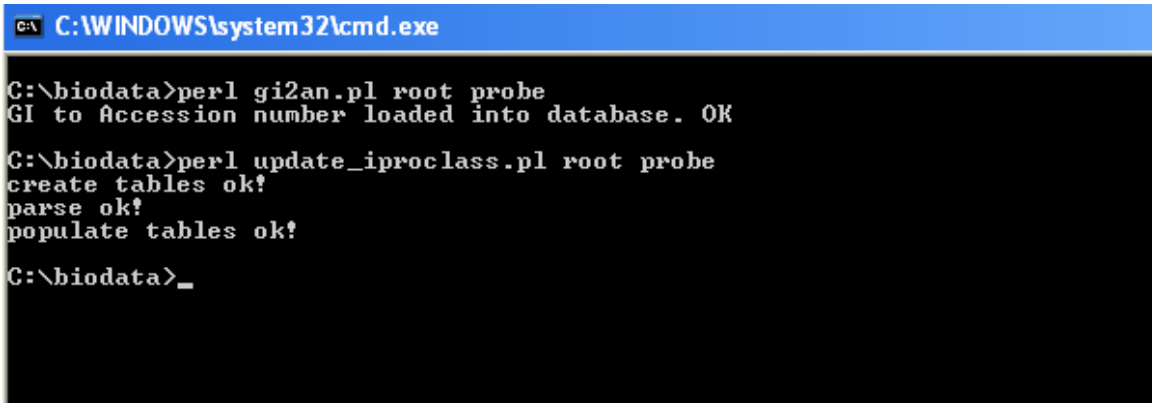
1. Τα περιεχόμενα των m_musculus.xml.gz και iproclass.tb.gz που προμηθευτήκαμε από τον διακομιστή **ftp://ftp.pir.georgetown.edu/databases/iproclass/.** Τα αρχεία αυτά βρίσκονται σε συμπιεσμένη μορφή και πρέπει πρώτα να αποσυμπιεστούν (τα Winzip, Winrar, 7zip είναι εφαρμογές που αποσυμπιέζουν αρχεία, με το τελευταίο να είναι δωρεάν και ανοιχτού κώδικα)
2. Τα script **update_iproclass.pl, gi2an.pl, xmlparser.pl** από το συνοδευτικό cd-rom.

Ο φάκελος θα πρέπει να έχει αυτή τη μορφή

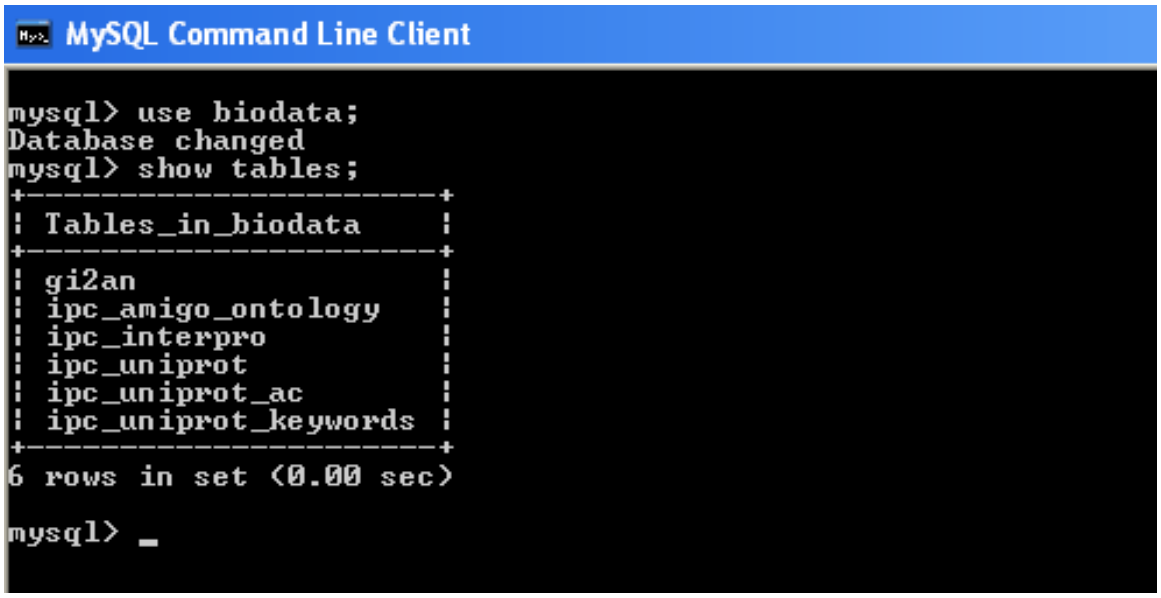


Επιστρέφουμε στην κονσόλα και εισάγουμε τις εξής εντολές

1. *perl gi2an.pl iproclass.tb root probe*
2. *perl update_iproclass.pl root probe*



Όπως βλέπουμε, στη βάση δεδομένων έχουν ήδη δημιουργηθεί οι νέοι πίνακες με τα δεδομένα που χρειαζόμαστε.



Εδώ έχει ολοκληρωθεί η εγκατάσταση της βάσης και η εφαρμογή πλέον μπορεί να χρησιμοποιηθεί μέσω ενός web browser, από τη διεύθυνση **http://localhost**

3.5 Ανανέωση των δεδομένων της βάση δεδομένων

Για ανανέωση των δεδομένων της βάσης, επαναλαμβάνουμε τα βήματα της ενότητας 3.4

4. ΣΤΟΧΟΙ ΓΙΑ ΤΟ ΜΕΛΛΟΝ

Η εφαρμογή έχει φτάσει σε ένα σημείο όπου μπορεί να χρησιμοποιηθεί για τον σκοπό για τον οποίο σχεδιάστηκε, δηλαδή να παρουσιάζει ομαδοποιημένες βιβλιογραφικές πληροφορίες για ένα σύνολο πρωτεϊνών. Επόμενοι στόχοι είναι

- προσθήκη φίλτρων που θα ορίζονται από το χρήστη, ώστε να μπορεί να μειώνει τον αριθμό των αποτελεσμάτων, με βάση κάποιες λέξεις κλειδιά
- προσθήκη πληροφοριών που έχουν σχέση με την αλληλεπίδραση μεταξύ των πρωτεϊνών, όπως οι βάσεις INTACT, BIND, DIP.
- παραμετροποίηση του τρόπου παρουσίασης των αποτελεσμάτων.

ΒΙΒΛΙΟΓΡΑΦΙΑ

- **Efficient biotinylation and single-step purification of tagged transcription factors in mammalian cells and transgenic mice**, by Ernie de Boer, Patrick Rodriguez, Edgar Bonte, Jeroen Krijgsveld, Eleni Katsantoni, Albert Heck, Frank Grosveld and John Strouboulis. Proc Natl Acad Sci U S A, Vol.100, no 13: 7480-7485, 2003.
- **Developing Bioinformatics Computer Skills**, by Cynthia Gibas, Per Jambeck, Published 2001 O'Reilly
- **Learning Perl**, by Randal L. Schwartz, Tom Phoenix, Brian D. Foy, Published 2005 O'Reilly
- **Programming Perl**, by Larry Wall, Tom Christiansen, Jon Orwant, Published 2000 O'Reilly
- **XML and Perl**, By Mark Riehl, Ilya Sterin, Published 2002, Sams Publishing
- **CGI Programming with Perl**, by Scott Guelich, Shishir Gundavaram, Gunther Birznieks, Published 2000, O'Reilly
- **Proteomics**, by Timothy Palzkill, ©2002 Kluwer Academic Publishers
- **Introduction to Proteomics, Tools for the New Biology**, by Daniel C. Liebler, Published 2002, Humana Press
- **Fundamentals of Database Systems**, by R. Elmasri, S.B Navathe, Published 2000, Addison-Wesley Publishing Company
- **http://www.ebi.uniprot.org/**
- **http://amigo.geneontology.org/**
- **http://www.ebi.ac.uk/interpro/**
- **http://pir.georgetown.edu/iproclass/**

ΠΑΡΑΡΤΗΜΑ Ά - ΚΩΔΙΚΑΣ

ΠΑ.1 gi2an.pl

```
#!/usr/bin/perl


##############################################################################################
###########
# Authors: Sakkos Nikolaos
# Last modified: July 18, 2007
##############################################################################################
###########



use DBI;


if(scalar @ARGV < 3){
       print "USAGE: perl gi2an.pl <file> <dbuser> <dbpassword> <dbname>\n";
}
$file = $ARGV[0];
$dbuser = $ARGV[1];
$dbpwd = $ARGV[2];
$dbname = $ARGV[3];


my ($Uniprot_ac,$Uniprot_id,$EntrezGene,$RefSeq,$GIID);
my @GIID_array;
my $counter;
```

```
#open connection to Access database

#my $dbh = DBI->connect('dbi:mysql:test','root','nyarlathotep');

# Prepare database connection

$dbh = DBI->connect("dbi:mysql:host=localhost", $dbuser, $dbpwd) or die "Can't connect with database $DBI::errstr\n";

$dbh->do("USE $dbname") or die "Error: $dbh->errstr";


$dbh->do("DROP TABLE IF EXISTS ");

$dbh->do("CREATE TABLE IF NOT EXISTS gi2an (GIID VARCHAR(15), AN VARCHAR (11) KEY)");


open (GI2AN, $file) or die "Could not open file $file\n";


while (GI2AN){

        ($Uniprot_ac,$Uniprot_id,$EntrezGene,$RefSeq,$GIID) = split("\t", $_, 6);

        @GIID_array = split("; ",$GIID);

        foreach (@GIID_array){

            my $sqlstatement="insert into gi2an (GIID,AN) VALUES ($_,'$Uniprot_ac')";

            my $sth = $dbh->prepare($sqlstatement);

            $sth->execute ||

                die "Could not execute SQL statement ... maybe invalid?";


            print "$_ is $Uniprot_ac\n";

        };

             $counter=$counter+1;

             print "$counter\n";;

    };

close (GI2AN);
```

ΠΑ.2 update_iproclass.pl

```
#!/usr/bin/perl


################################################################################################
############

# Authors: Sakkos Nikolaos

# Last modified: August 18, 2007

################################################################################################
############




use DBI();

use Cwd;




if(scalar @ARGV < 3){

       print "USAGE: perl update_iproclass.pl <dbuser> <dbpassword> <dbname>\n";

}


$dbuser = $ARGV[0];

$dbpwd = $ARGV[1];

$dbname = $ARGV[2];


# Prepare database connection

$dbh = DBI->connect("dbi:mysql:host=localhost", $dbuser, $dbpwd) or die "Can't connect
with database $DBI::errstr\n";

$dbh->do("USE $dbname") or die "Error: $dbh->errstr";
```

```
createTables();

print "create tables ok!\n";

#parse();

print "parse ok!\n";

populateTables();

print "populate tables ok!\n";



################################## Create tables
##############################################################################################


sub createTables {


$dbh->do("DROP TABLE IF EXISTS ipc_uniprot");

$dbh->do("DROP TABLE IF EXISTS ipc_ontology");

$dbh->do("DROP TABLE IF EXISTS ipc_uniprot_ac");

$dbh->do("DROP TABLE IF EXISTS ipc_uniprot_keywords");

$dbh->do("DROP TABLE IF EXISTS ipc_interpro");


$dbh->do("CREATE TABLE IF NOT EXISTS ipc_uniprot (ipcid VARCHAR(15) KEY, uniprotid
VARCHAR(11), uniprot_source VARCHAR(20), uniprot_name VARCHAR(200), uniprot_function
VARCHAR(2500))");

$dbh->do("CREATE TABLE IF NOT EXISTS ipc_uniprot_keywords (ipcid VARCHAR(15) KEY,
keywords VARCHAR(50))");

$dbh->do("CREATE TABLE IF NOT EXISTS ipc_uniprot_ac (ipcid VARCHAR(15), accession_number
VARCHAR (15) KEY)");

$dbh->do("CREATE TABLE IF NOT EXISTS ipc_amigo_ontology (ipcid VARCHAR(15) KEY, functions
VARCHAR (200))");
```

```
$dbh->do("CREATE TABLE IF NOT EXISTS ipc_interpro (ipcid VARCHAR(15) KEY, interproid VARCHAR(9), interprodesc VARCHAR(100))");

}

################################## Parse iProClass.xml ##############################################################################

#sub parse {

#system("rmdir /S /Q ipc");

#system("mkdir ipc");

#print "Downloading iproclass.xml.gz...\n";

#system("wget ftp://ftp.pir.georgetown.edu/pir_databases/iproclass/iproclass.xml.gz --directory-prefix=ipc/") == 0 or die "Error: $?\n";

#print "\ndone downloading iproclass.xml.gz\n";

#print "Decompressing... ";

#system("gzip -d  ipc/iproclass.xml.gz") == 0 or die "$?\n";

#print "done.\n";


#system("perl xmlparser.pl ipc/iproclass.xml");

#}

################################ Load parsed data into tables ##############################################################################


sub populateTables {

print "Loading data into ipc_uniprot...\n";

$fullFilePath = getcwd()."/ipc/parsed/ipc_uniprot.txt ";

$dbh->do("LOAD DATA LOCAL INFILE \'${fullFilePath}\' IGNORE INTO TABLE ipc_uniprot");


print "Loading data into ipc_uniprot_keywords...\n";

$fullFilePath = getcwd()."/ipc/parsed/ipc_uniprot_keywords.txt";
```

```
$dbh->do("LOAD DATA LOCAL INFILE \'${fullFilePath}\' IGNORE INTO TABLE ipc_uniprot_keywords");


print "Loading data into ipc_uniprot_ac...\n";

$fullFilePath = getcwd()."/ipc/parsed/ipc_uniprot_ac.txt";

$dbh->do("LOAD DATA LOCAL INFILE \'${fullFilePath}\' IGNORE INTO TABLE ipc_uniprot_ac");


print "Loading data into ipc_amigo_ontology...\n";

$fullFilePath = getcwd()."/ipc/parsed/ipc_amigo_ontology.txt";

$dbh->do("LOAD DATA LOCAL INFILE \'${fullFilePath}\' IGNORE INTO TABLE ipc_amigo_ontology");


print "Loading data into ipc_interpro...\n";

$fullFilePath = getcwd()."/ipc/parsed/ipc_interpro.txt";

$dbh->do("LOAD DATA LOCAL INFILE \'${fullFilePath}\' IGNORE INTO TABLE ipc_interpro");


print "done loading data into tables from iProClass.\n";

}
```

ΠΑ.3 xmlparser.pl

```
#!/usr/bin/perl


##############################################################################################
###########

# Authors: Sakkos Nikolaos

# Last modified: July 18, 2007

# Requires XML::Parser::Expat Perl module from CPAN

##############################################################################################
###########


use XML::Parser::Expat;


if(scalar(@ARGV) <= 0){

       print "USAGE: perl xmlparser.pl <iproclass.xml location>";

       die;

}


$xmlFile = $ARGV[0];


$parser = new XML::Parser::Expat;

$parser->setHandlers('Start' => \&startXML,'End'   => \&endXML, 'Char'=>\&charNothing);


open (XML, $xmlFile) or die "Could not open file $xmlFile\n";


system("rm -r ipc/parsed");
```

```
system("mkdir ipc/parsed");


open (UNIPROTAC, ">./ipc/parsed/ipc_uniprot_ac.txt") or die "Could not create ./ipc/parsed/ipc_uniprot_ac.txt\n";

open (UNIPROT, ">./ipc/parsed/ipc_uniprot.txt") or die "Could not create ./ipc/parsed/ipc_uniprot.txt\n";

open (ONT, ">./ipc/parsed/ipc_amigo_ontology.txt") or die "Could not create ./ipc/parsed/ipc_amgigo_ontology.txt\n";

#open (FUNC, ">./ipc/parsed/ipc_uniprot_function.txt") or die "Could not create ./ipc/parsed/ipc_uniprot_function.txt\n";

open (KEY, ">./ipc/parsed/ipc_uniprot_keywords.txt") or die "Could not create ./ipc/parsed/ipc_uniprot_keywords.txt\n";

open (INTERPRO, ">./ipc/parsed/ipc_interpro.txt") or die "Could not create ./ipc/parsed/ipc_interpro.txt\n";


# Variables

my $entryid;

my %data;

my $entryNum = 0;

my $record;                     #buffer for elements bigger than a line

my $context;                    #variable where the name of the element is being stored as a global variable


$parser->parse(*XML);


#foreach (keys %data) {print "$_ is $data{$_}\n"};


close(UNIPROT);

close(ONT);
```

```
#close(FUNC);
close(KEY);
close(INTERPRO);

sub startXML {

	my($p, $el, %atts) = @_;

	$context= $el;
	$record = {} if ( $el eq 'iProClassEntry');

	$p->setHandlers('Char'=>\&charNothing);

	if($el eq 'iProClassEntry'){

		# New entry.
		$entryid = $atts{'Entry_ID'};
		$entryNum++;
		print "$entryid\t$entryNum\n";

	}elsif($el eq 'UniProtKB_ID'){

		$p->setHandlers('Char'	=>	\&charUniProtId);

	}elsif($el eq 'UniProtKB_Accession'){
```

```
        $p->setHandlers('Char'     => \&charUniProtAc);

}elsif($el eq 'Source_Organism'){

        $p->setHandlers('Char'     => \&charUniProtSource);

}elsif($el eq 'Protein_Name'){

        $p->setHandlers('Char'     => \&charUniProtName);

}elsif($el eq 'Function_Info'){

        $p->setHandlers('Char' => \&charFunction);




}elsif($el eq 'GO_Term'){

        $p->setHandlers('Char'=>\&charOntology);

}elsif($el eq 'keyword'){

        $p->setHandlers('Char'=>\&charKeywords);

}elsif($el eq 'InterPro_ID'){
```

```
                $p->setHandlers('Char'=>\&charInterproId);


        }elsif($el eq 'InterPro_Desc'){


                $p->setHandlers('Char'=>\&charInterproDesc);
        }
}


sub charNothing{}


sub charInterproId {
        my($p, $str) = @_;
        $data{'InterPro_ID'} = $str;




}


sub charInterproDesc {
        my($p, $str) = @_;
        $data{'InterPro_Desc'} = $str;


}




sub charKeywords {
        my($p,$str) = @_;
```

```
	$data{'keyword'} = $str;


}

sub charOntology {
	my($p,$str)=@_;
	$data{'GO_Term'}=$str;


}



sub charFunction {
	my($p,$str)=@_;
	$record->{ $context } .= $str;
	#$data{'Function_Info'}=$str;






}

sub charUniProtId {
	my($p, $str) = @_;
	$data{'UniProtKB_ID'} = $str;
```

```
}

sub charUniProtAc {

        my($p,$str) = @_;
        $data{'UniProtKB_Accession'} = $str;
        #my $line = $p->current_line;
        #print "I'm working on line $line\n";
}

sub charUniProtSource {

        my($p,$str) = @_;
        $data{'Source_Organism'} = $str;
        #print "$data{'Source_Organism'}\n"

}

sub charUniProtName {

        my($p,$str)=@_;
        $record->{ $context } .= $str;

}

sub endXML {
```

```
	my($p, $el) = @_;


	$p->setHandlers('Char'=>\&charNothing);


	if($el eq 'UniProtKB_Accession'){


		#print UNIPROTAC "$entryid\t$data{'UniProtKB_ID'}\t$data{'UniProtKB_Accession'}\n";

		print UNIPROTAC "$entryid\t$data{'UniProtKB_Accession'}\n";


	}elsif ($el eq 'Function_Info'){

		$data{'Protein_Name'} = $record->{'Protein_Name'};		#the protein_name text is being saved with trailing and leading \n

		$data{'Protein_Name'} =~ s/\n//g;

		$data{'Function_Info'} = $record->{'Function_Info'};		#the function_info text is being saved with trailing and leading \n

		$data{'Function_Info'} =~ s/\n//g;				#replace \n with void


		print UNIPROT "$entryid\t$data{'UniProtKB_ID'}\t$data{'Source_Organism'}\t$data{'Protein_Name'}\t$data{'Function_Info'}\n";



	}elsif($el eq 'GO_Term'){


		print ONT "$entryid\t$data{'GO_Term'}\n";
```

```
	}elsif($el eq 'iProClassEntry'){


		undef $data;

		$entryid = "";


	}elsif($el eq 'keyword'){


		print KEY "$entryid\t$data{'keyword'}\n";




	}elsif($el eq 'InterPro_Desc'){


		print INTERPRO "$entryid\t$data{'InterPro_ID'}\t$data{'InterPro_Desc'}\n";


	}


}
```

П.А5 index.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

</head>




<body bgcolor="#cccccc">









<a href="/"><img style="border: 0px solid ; width: 800px; height: 130px;" alt="banner"
src="protein-banner-illustration.jpg"></a><br>









<br>









<br>
```

```
<span style="font-weight: bold;">Welcome to P.S.M.E! <br>

This search tool provides compact information about proteins of interest, sorted according to score or hits.<br>

<br>

Submit your mass spectrometry results through the form below. Press the

"Submit File" button when ready to be redirected<br>

to your results.<br>

</span><br>
```

```
<form action="upload.cgi" method="post" enctype="multipart/form-data"><br>



File to Upload: <input name="excel" type="file">


      Type of File:


  <select name="type">

  <option value="excel1">Excel File</option>

  <option value="excel2">Excel File - GI</option>

  <option value="text1">Text File with GI numbers</option>


  </select>


  <br>

  <br>

  <input name="Submit" value="Submit File" type="submit">




</form>
```

```
<br>

<small><br>

Attention!<br>

Be sure to change your browser settings according to this <a target="_blank" href="/settings.png">image</a></small>,<small> before starting uploading your results.</small><br>

<br>

<br>
```



```
<br>

<br>

<br>

<small>copyright 2007 by Nicholas Sakkos&amp;nbsp;</small>

</body>

</html>
```

Π.Α5 upload.cgi

```
#!/perl/bin/perl


use CGI qw(:standard :html3);

use CGI::Carp qw(warningsToBrowser fatalsToBrowser);

use Win32::OLE qw(in with);

use Win32::OLE::Const 'Microsoft Excel';

use DBI;

use Archive::Zip qw( :ERROR_CODES :CONSTANTS );









$upload_dir = "C:/Program Files/Apache Software Foundation/Apache2.2/htdocs/upload";

$results_dir =  "C:/Program Files/Apache Software Foundation/Apache2.2/htdocs/results";

$query = new CGI;

$filename = $query->param("excel");

$type = $query->param("type");

$filename =~ s/.*[\/\\](.*)/$1/;                    #eliminates file path and leaves file
name

$upload_filehandle = $query->upload("excel");




$uniprot_url='http://www.pir.uniprot.org/cgi-bin/upEntry?id=';

$ncbi_url='http://www.ncbi.nlm.nih.gov/entrez/viewer.fcgi?db=protein&id=';

$iproclass_url='http://pir.georgetown.edu/cgi-bin/ipcEntry?id=';
```

```
$mgi_url_start='http://www.informatics.jax.org/javawi2/servlet/WIFetch?page=searchTool&query=';

$mgi_url_end='&selectedQuery=Accession+IDs';

$ensembl_url='http://www.ensembl.org/Mus_musculus/searchview?species=Mus_musculus;idx=;q=';


system("rmdir /S /Q \"$results_dir\"");

system("mkdir \"$results_dir\"");


my @worksheet_names;


open UPLOADFILE, ">$upload_dir/$filename";


binmode UPLOADFILE;                                                    # ensures the transfers uses binary mode, so that the file is not altered under different os circumstances


while ( <$upload_filehandle> ) {

       print UPLOADFILE;

} ;


close UPLOADFILE;


#prints the banner as well as some text before the results

print

       $query->header .

       $query->start_html(-title=>'Results', -bgcolor=>"#CCCCCC") .
```

```
	'<a href="/"><img style="width: 800px; height: 130px;" alt="banner" src="protein-banner-illustration.jpg"></a><br>';




my @all_data;			#Contains the hashes. Each hash contains the data of a single worksheet

my $list;				#reference to a hash that contains the proteins. each list represents one worksheet



my $xlsFile;

my $Excel;

my $Book;

my $sheetcnt;



if ($type eq "excel1"){

	&excel_file_1;

};

if ($type eq "text1"){

	&text_file_1;

};
```

```
if ($type eq "excel2"){


	&excel_file_2;

};

#print "\nkai poy lete to type einai " . $type ."\n";

print  $query->end_html;

exit (0);




###############

# the subroutines used  #

##############


sub text_file_1{


	my $txtfile = "$upload_dir/$filename";

	open TEXTFILE, $txtfile;


	while (<TEXTFILE>){

		if (/(gi\|\w+)\s/) {		# memorize the gi number

			$gi_value= $1;

			$gi_value=~ s/gi\|(.*)/$1/;

			$protein = {

				GI => $1,

				HIT => "-",
```

```
                                SCORE => "-",

                                EXCEL_NAME => "-"

                        };

                        $list->{$protein->{GI}} = $protein;


                };

        };

        push @all_data, $list;


        &retrieve_from_db(%{$all_data[0]});




        # foreach $k (keys %{$all_data[0]}){

                # $tester = $all_data[0]{$k}->{uniprot_name};

                # print $query->h3("the uniprot name was $tester\n");  # the word was there

        # };




        print '<br><br><span style="font-weight: bold;"><big>or upload a new file:
</big></span><br><br>';

        print '<form action="upload.cgi" method="post" enctype="multipart/form-data">

                        File to Upload: <input name="excel" type="file">


                              <br>Type of File:   


                          <select name="type">

                          <option value="excel1">Excel File</option>
```

```
                    <option value="excel2">Excel File - GI</option>

                    <option value="text1">Text File with GI numbers</option>

                    <option value="text2">Text type 2</option>

                    </select>




                    <input name="Submit" value="Submit File" type="submit">




                  </form>';

      print  '<table style="width: 800px;" class="boxTable" bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">';


      foreach $n (keys %{$all_data[0]}){

            print       '<tr><th  class="right"  colspan="2"  align="center">gi|'  . $all_data[0]{$n}->{GI};

            print      '<tr><td  bgcolor="#CC9999"  width="20%">Protein  Name  and Organism</td>';

            print  '<td width="80%"><i>Name: </i> ' . $all_data[0]{$n}->{uniprot_name} . '<br><br>';

            print  '<i>Organism:  </i>'  .  $all_data[0]{$n}->{uniprot_source}  . '<br></td>';


            print  '<tr><td bgcolor="#CCCCFF" width="20%">Keywords</td>';

            print      '<td  width="80%"><i>Keywords:  </i>  '  .      join(",  ", @{$all_data[0]{$n}->{keywords}}) . '<br></td>';
```

```
            print  '<tr><td bgcolor="#CCCCFF" width="20%">Function</td>';

            print       '<td       width="80%"><i>Function:       </i>       '       .
$all_data[0]{$n}->{uniprot_function} . '<br></td>';


            print  '<tr><td bgcolor="#CCCCFF" width="20%">Ontology</td>';

            print      '<td   width="80%"><i>Ontology:   </i>   '   .      join(",   ",
@{$all_data[0]{$n}->{functions}}) . '<br></td>';


            print  '<tr><td bgcolor="#CCCCFF" width="20%">Family Classification</td>';

            print     '<td   width="80%"><i>InterPro   :   </i>   '   .     join(",   ",
@{$all_data[0]{$n}->{interprodesc}}) . '<br></td>';


            print  '<tr><td bgcolor="#CCCCFF" width="20%">External Links</td>';

            print    '<td  width="80%"><i>NCBI  :  </i>  <a  href="'  .  $ncbi_url  .
$all_data[0]{$n}->{GI} . '" target="_blank">' . $all_data[0]{$n}->{GI} . '</a><br>';

            print       '<i>UniProt    :    </i>    <a    href="'    .    $uniprot_url    .
$all_data[0]{$n}->{uniprotid} . '" target="_blank">' . $all_data[0]{$n}->{uniprotid} .
'</a><br>';

            #print      '<i>iProClass   :   </i>   <a   href="'   .   $iproclass_url   .
$all_data[0]{$n}->{ipcid}   .   '"   target="_blank">'   .   $all_data[0]{$n}->{ipcid}   .
'</a><br>';

            print       '<i>MGI    :    </i>    <a    href="'    .    $mgi_url_start    .
$all_data[0]{$n}->{accession_number}   .   $mgi_url_end   .   '"   target="_blank">'   .
$all_data[0]{$n}->{accession_number} . '</a><br>';

            print       '<i>Ensembl    :    </i>    <a    href="'    .    $ensembl_url    .
$all_data[0]{$n}->{accession_number}        .        '"        target="_blank">'        .
$all_data[0]{$n}->{accession_number} . '</a><br></td>';


      };

      print  '</table>';
```

```
	print "\nyeeha\n";
	close TEXTFILE;
}

sub excel_file_1{

	print $query->h3("Use this table to navigate through the results" );

	$xlsFile = "$upload_dir/$filename";

	$Excel = Win32::OLE->GetActiveObject('Excel.Application')	#  get  already
active Excel application or open new
		|| Win32::OLE->new('Excel.Application', 'Quit');
    $Book      =      $Excel->Workbooks->Open("$xlsFile");
# open Excel file
```

```
   $sheetcnt      =      $Book->Worksheets->Count();
#find how many worksheets are in the file


       foreach  (1..$sheetcnt){

              $list=();                                        #blank the %list. Don't know
why, even though it's a  scalar, it needs to be initialized in this way

              &excel_filter($_);                               #retrieves   the   information
from the excel file and populates the %list hash

              push @all_data, $list;                           #save   the   data   into   the
all_data array. Each cell contains the data of each worksheet



              &retrieve_from_db(%{$all_data[$_-1]});
#retrieves extra information from the DB and fills in the %list hash


       };


       $filename =~ s/\.xls//;


       foreach  (1..$sheetcnt){




              &print_summary(%{$all_data[$_-1]});

              &print_details(%{$all_data[$_-1]});


       };
```

```
	&archive_files;


	&table_menu;

	print '<br><br><span style="font-weight: bold;"><big>or upload a new file:
</big></span><br><br>';

	print '<form action="upload.cgi" method="post" enctype="multipart/form-data">

		File to Upload: <input name="excel" type="file">


		      <br>Type of File:   


		  <select name="type">

		  <option value="excel1">Excel File</option>

		  <option value="excel2">Excel File - GI</option>

		  <option value="text1">Text File with GI numbers</option>

		  <option value="text2">Text type 2</option>

		  </select>




		  <input name="Submit" value="Submit File" type="submit">




		</form>';

}
```

```
sub excel_file_2{

       $xlsFile = "$upload_dir/$filename";

       $Excel = Win32::OLE->GetActiveObject('Excel.Application')          #   get   already
active Excel application or open new
              || Win32::OLE->new('Excel.Application', 'Quit');
   $Book      =      $Excel->Workbooks->Open("$xlsFile");
# open Excel file

       my $Sheet = $Book->Worksheets(1);
       my                                    $LastRow                                    =
$Sheet->UsedRange->Find({What=>"*",SearchDirection=>xlPrevious,SearchOrder=>xlByRows})->{
Row};

       foreach (1..$LastRow){

       $gi_value = $Sheet->Cells($_,1)->{'Value'};
              if ($gi_value =~ m/(gi\|\w+)/) {             # memorize the gi number
                     $gi_value= $1;
                     $gi_value=~ s/gi\|(.*)/$1/;
                     $protein = {
                                   GI => $1,
                                   HIT => "-",
                                   SCORE => "-",
```

```
                                EXCEL_NAME => "-"

                };

                $list->{$protein->{GI}} = $protein;


        };
};
push @all_data, $list;


&retrieve_from_db(%{$all_data[0]});



# foreach $k (keys %{$all_data[0]}){
        # $tester = $all_data[0]{$k}->{uniprot_name};
        # print $query->h3("the uniprot name was $tester\n");  # the word was there
# };



print '<br><br><span style="font-weight: bold;"><big>or upload a new file:
</big></span><br><br>';
print '<form action="upload.cgi" method="post" enctype="multipart/form-data">
                File to Upload: <input name="excel" type="file">


                      <br>Type of File:   


                  <select name="type">
                  <option value="excel1">Excel File</option>
                  <option value="excel2">Excel File - GI</option>
```

```
                    <option value="text1">Text File with GI numbers</option>

                    <option value="text2">Text type 2</option>

                    </select>




                    <input name="Submit" value="Submit File" type="submit">




                  </form>';

      print  '<table style="width: 800px;" class="boxTable" bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">';


      foreach $n (keys %{$all_data[0]}){

            print       '<tr><th  class="right"  colspan="2"  align="center">gi|'  . $all_data[0]{$n}->{GI};

            print      '<tr><td  bgcolor="#CC9999"  width="20%">Protein  Name  and Organism</td>';

            print  '<td width="80%"><i>Name: </i> ' . $all_data[0]{$n}->{uniprot_name} . '<br><br>';

            print  '<i>Organism:  </i>'  .  $all_data[0]{$n}->{uniprot_source}  . '<br></td>';


            print  '<tr><td bgcolor="#CCCCFF" width="20%">Keywords</td>';

            print      '<td  width="80%"><i>Keywords:  </i>  '  .      join(",  ", @{$all_data[0]{$n}->{keywords}}) . '<br></td>';
```

```
            print  '<tr><td bgcolor="#CCCCFF" width="20%">Function</td>';

            print       '<td       width="80%"><i>Function:       </i>       '       .
$all_data[0]{$n}->{uniprot_function} . '<br></td>';


            print  '<tr><td bgcolor="#CCCCFF" width="20%">Ontology</td>';

            print      '<td   width="80%"><i>Ontology:   </i>   '   .      join(",   ",
@{$all_data[0]{$n}->{functions}}) . '<br></td>';


            print  '<tr><td bgcolor="#CCCCFF" width="20%">Family Classification</td>';

            print     '<td   width="80%"><i>InterPro   :   </i>   '   .     join(",   ",
@{$all_data[0]{$n}->{interprodesc}}) . '<br></td>';


            print  '<tr><td bgcolor="#CCCCFF" width="20%">External Links</td>';

            print    '<td  width="80%"><i>NCBI  :  </i>  <a  href="'  .  $ncbi_url  .
$all_data[0]{$n}->{GI} . '" target="_blank">' . $all_data[0]{$n}->{GI} . '</a><br>';

            print       '<i>UniProt    :    </i>    <a    href="'    .    $uniprot_url    .
$all_data[0]{$n}->{uniprotid} . '" target="_blank">' . $all_data[0]{$n}->{uniprotid} .
'</a><br>';

            #print      '<i>iProClass   :   </i>   <a   href="'   .   $iproclass_url   .
$all_data[0]{$n}->{ipcid}   .   '"   target="_blank">'   .   $all_data[0]{$n}->{ipcid}   .
'</a><br>';

            print       '<i>MGI    :    </i>    <a    href="'    .    $mgi_url_start    .
$all_data[0]{$n}->{accession_number}   .   $mgi_url_end   .   '"   target="_blank">'   .
$all_data[0]{$n}->{accession_number} . '</a><br>';

            print       '<i>Ensembl    :    </i>    <a    href="'    .    $ensembl_url    .
$all_data[0]{$n}->{accession_number}        .        '"        target="_blank">'        .
$all_data[0]{$n}->{accession_number} . '</a><br></td>';


      };

      print  '</table>';
```

```
}


sub excel_filter{
##################################
# opens the excel file, partially populates the record so
# that we can start querying the databases. Also prints the
# results we found
##################################



       my ($n) = (@_);
       my $Sheet = $Book->Worksheets($n);


       my $row=4;


       if (defined $Sheet->Cells($row,6)->{'Value'}){
              push @worksheet_names, $Book->Worksheets($_)->{Name};
       };



       while (defined $Sheet->Cells($row,6)->{'Value'}) {
                     unless              ($Sheet->Cells($row,3)->{'Value'}              eq
$Sheet->Cells($row,7)->{'Value'}){
```

```
                        $gi_value = $Sheet->Cells($row,7)->{'Value'};
                        $gi_value=~ s/gi\|(.*)/$1/;
                        $hit_value= $Sheet->Cells($row,5)->{'Value'};
                        $score_value= $Sheet->Cells($row,6)->{'Value'};
                        $excel_name= $Sheet->Cells($row,8)->{'Value'} ;

                        $protein = {
                               GI => $gi_value,
                               HIT => $hit_value,
                               SCORE => $score_value,
                               EXCEL_NAME => $excel_name
                        };
                        $list->{$protein->{GI}} = $protein;
                };
                $row+=1;

        };




        # clean up the excel thing
        #$Book->Close;

}

sub by_score{
```

```
	$list_to_print{$b}->{SCORE} <=> $list_to_print{$a}->{SCORE}

}	#sorts results by score


sub retrieve_from_db{


	my (%list_to_retrieve) = @_;


	$dbh	=	DBI->connect('dbi:mysql:database=biodata;host=localhost',	'root',
'nyarlathotep',

		{ RaiseError => 1, AutoCommit => 1});


	foreach $k (keys %list_to_retrieve){


		$sth = $dbh->prepare("SELECT AN from gi2an WHERE GIID=?");

		$sth->execute($k);


		while (@row = $sth->fetchrow_array) {


			$h = $row[0];

			$h =~ s/^\s+//;

			$h =~ s/\s+$//;

			$list_to_retrieve{$k}->{accession_number} = $h;

			last;

		};
```

```
		$sth = $dbh->prepare("SELECT ipcid from ipc_uniprot_ac WHERE accession_number=?");
		$sth->execute($h) or die $sth->errstr;

		while (@row = $sth->fetchrow_array) {

			$h = $row[0];
			$h =~ s/^\s+//;
			$h =~ s/\s+$//;

			$list_to_retrieve{$k}->{ipcid} = $h;
		};

		$sth = $dbh->prepare("SELECT uniprotid, uniprot_source, uniprot_name, uniprot_function from ipc_uniprot WHERE ipcid=?");
		$sth->execute($h) or die $sth->errstr;

		while (@row = $sth->fetchrow_array) {
			$list_to_retrieve{$k}->{uniprotid} = $row[0];
			$list_to_retrieve{$k}->{uniprot_source} = $row[1];
			$list_to_retrieve{$k}->{uniprot_name} = $row[2];
			$list_to_retrieve{$k}->{uniprot_function} = $row[3];
		};

		$sth = $dbh->prepare("SELECT keywords FROM ipc_uniprot_keywords WHERE ipcid=?");
		$sth->execute($h) or die $sth->errstr;
```

```
while (@row = $sth->fetchrow_array){

       push @{$list_to_retrieve{$k}->{keywords}}, $row[0];

};

$sth = $dbh->prepare("SELECT functions FROM ipc_amigo_ontology WHERE
ipcid=?");
$sth->execute($h) or die $sth->errstr;



while (@row = $sth->fetchrow_array){
       $m = $row[0];
       $m =~ s/\s+$//;            #the words have trailing backslashes

       push @{$list_to_retrieve{$k}->{functions}}, $row[0];

};

$sth = $dbh->prepare("SELECT interproid,interprodesc FROM ipc_interpro
WHERE ipcid=?");
$sth->execute($h) or die $sth->errstr;

while (@row = $sth->fetchrow_array){

       push @{$list_to_retrieve{$k}->{interproid}}, $row[0];
       push @{$list_to_retrieve{$k}->{interprodesc}}, $row[1];
```

```
            };

};

                #$record = {
                #1     GI => excel
                #1     HIT => excel
                #1     SCORE =>  excel
                #4     uniprotid => ipc_uniprot
                #4     uniprot_source => ipc_uniprot
                #4     uniprot_name  => ipc_uniprot
                #4     uniprot_function => ipc_uniprot
                #3     ipcid => ipc_uniprot_ac
                #2     accession_number => gi2an
                #5     keywords => ipc_uniprot_keywords
                #6     functions => ipc_amigo_ontology
                #7     interproid => ipc_interpro
                #7     interprodesc=> ipc_interpro
                #}

}
```

```
sub print_details{

	my (%list_to_print) = @_;

	if ((keys %list_to_print)==0) { return (0)};  #if the list is empty, which means that the worksheet is empty, do nothing



	my $link_name = $worksheet_names[$_-1] . "_details_score.html";

	open SUMMARY, ">$results_dir/$link_name";  # or die "Could not create $results_dir/$link_name\n";

	print SUMMARY '<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" xml:lang="en-US"><head><title>' .

				$Book->Worksheets($_)->{Name} .

				' Details sorted by score</title>

				</head><body bgcolor="#CCCCCC"><a href="/"><img style="width: 800px; height: 130px;" alt="banner" src="../protein-banner-illustration.jpg"></a><br>';



	print SUMMARY '<h3>Use this table to navigate through the results</h3>';

	print SUMMARY '<TABLE style="width: 500px;" class="boxTable" bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">' . "\n";

	print SUMMARY "<tr><th>#</th><th>Worksheet</th><th>Summary</th><th>Details</th></tr>\n";

	$worksheet_number = 0;

	foreach (@worksheet_names)  {

		$worksheet_number+=1;

		print SUMMARY "<tr><td>" .

				$worksheet_number .

				'</td><td style="text-align: center;">' .
```

```
                        $_  .

                        '</td><td style="text-align: center;">' .

                        '<a    href="/results/'.    $_    .    '_summary_score.html">by
score</a>    <a href="/results/' . $_ . '_summary_hits.html">by hits</a><br>' .

                        '</td><td style="text-align: center;">' .

                        '<a    href="/results/'.    $_    .    '_details_score.html">by
score</a>    <a href="/results/' . $_ . '_details_hits.html">by hits</a><br>' .

                        "</td></tr>";

      };


      #<a    href="'    .    $uniprot_url    .    $list_to_print{$n}->{uniprotid}    .    '"
target="_blank">' . $list_to_print{$n}->{uniprotid} . '</a><br>';

      # $link_name = "worksheet" . $_ . "_details.html";

      print  SUMMARY "</TABLE>\n";

      print  SUMMARY  '<TABLE  style="width:  500px;"  class="boxTable"  bgcolor="#ffffff"
border="1" cellpadding="8" cellspacing="0">' . "\n";

      print SUMMARY '<tbody>

                        <tr>

                        <td style="text-align: center;"><a href="/results/'. $filename
. '.zip" target="_blank">Download the results</a></td>

                    </tr>

                    </tbody>';

      print SUMMARY '</TABLE>';


      print SUMMARY '<br><br><span style="font-weight: bold;"><big>or upload a new file:
</big></span><br><br>';

      print       SUMMARY       '<form       action="../upload.cgi"       method="post"
enctype="multipart/form-data">

                    File to Upload: <input name="excel" type="file">
```

```
    <br>Type of File:   


<select name="type">

<option value="excel1">Excel File</option>

<option value="excel2">Excel File - GI</option>

<option value="text1">Text File with GI numbers</option>

<option value="text2">Text type 2</option>

</select>




<input name="Submit" value="Submit File" type="submit">




</form>';
```

```
print SUMMARY "<br><h3>This is worksheet #$_, called <i>\"" . $Book->Worksheets($_)->{Name} ."\"</i></h3>";

print SUMMARY $query->h3("A detailed view of the results sorted by score follows<br>");

print SUMMARY '<table style="width: 800px;" class="boxTable" bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">';
```

```
      foreach $n (sort {$list_to_print{$b}->{SCORE} <=> $list_to_print{$a}->{SCORE}}
keys %list_to_print){

            print SUMMARY '<tr><th class="right" colspan="2" align="left">gi|' .
$list_to_print{$n}->{GI} . ' has ' . $list_to_print{$n}->{HIT} . ' hits and a ' .
$list_to_print{$n}->{SCORE} . ' score</th></tr>';

            print SUMMARY '<tr><td bgcolor="#CC9999" width="20%">Protein Name and
Organism</td>';

            print SUMMARY '<td width="80%"><i>Name: </i> ' .
$list_to_print{$n}->{uniprot_name} . '<br><br>';

            print SUMMARY '<i>Organism: </i>' . $list_to_print{$n}->{uniprot_source} .
'<br></td>';


            print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">Keywords</td>';

            print SUMMARY '<td width="80%"><i>Keywords: </i> ' . join(", ",
@{$list_to_print{$n}->{keywords}}) . '<br></td>';


            print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">Function</td>';

            print SUMMARY '<td width="80%"><i>Function: </i> ' .
$list_to_print{$n}->{uniprot_function} . '<br></td>';


            print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">Ontology</td>';

            print SUMMARY '<td width="80%"><i>Ontology: </i> ' . join(", ",
@{$list_to_print{$n}->{functions}}) . '<br></td>';


            print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">Family
Classification</td>';

            print SUMMARY '<td width="80%"><i>InterPro : </i> ' . join(", ",
@{$list_to_print{$n}->{interprodesc}}) . '<br></td>';


            print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">External Links</td>';
```

```
            print SUMMARY '<td width="80%"><i>NCBI : </i> <a href="' . $ncbi_url .
$list_to_print{$n}->{GI} . '" target="_blank">' . $list_to_print{$n}->{GI} . '</a><br>';

            print SUMMARY '<i>UniProt : </i> <a href="' . $uniprot_url .
$list_to_print{$n}->{uniprotid} . '" target="_blank">' . $list_to_print{$n}->{uniprotid}
. '</a><br>';

            #print SUMMARY '<i>iProClass : </i> <a href="' . $iproclass_url .
$list_to_print{$n}->{ipcid} . '" target="_blank">' . $list_to_print{$n}->{ipcid} .
'</a><br>';

            print SUMMARY '<i>MGI : </i> <a href="' . $mgi_url_start .
$list_to_print{$n}->{accession_number} . $mgi_url_end . '" target="_blank">' .
$list_to_print{$n}->{accession_number} . '</a><br>';

            print SUMMARY '<i>Ensembl : </i> <a href="' . $ensembl_url .
$list_to_print{$n}->{accession_number} . '" target="_blank">' .
$list_to_print{$n}->{accession_number} . '</a><br></td>';


      };

      print SUMMARY '</table>';

      print SUMMARY $query->end_html;

      close (SUMMARY);


      my $link_name = $worksheet_names[$_-1] . "_details_hits.html";

      open SUMMARY, ">$results_dir/$link_name"; # or die "Could not create
$results_dir/$link_name\n";

      print SUMMARY '<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US"
xml:lang="en-US"><head><title>' .

                        $Book->Worksheets($_)->{Name} .

                        ' Details sorted by hits</title>

                        </head><body bgcolor="#CCCCCC"><a href="/"><img
style="width: 800px; height: 130px;" alt="banner"
src="../protein-banner-illustration.jpg"></a><br>';
```

```
	print SUMMARY '<h3>Use this table to navigate through the results</h3>';

	print SUMMARY '<TABLE style="width: 500px;" class="boxTable" bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">' . "\n";

	print SUMMARY "<tr><th>#</th><th>Worksheet</th><th>Summary</th><th>Details</th></tr>\n";

	$worksheet_number = 0;


	foreach (@worksheet_names) {

		$worksheet_number+=1;

		print SUMMARY "<tr><td>" .

				$worksheet_number .

				'</td><td style="text-align: center;">' .

				$_ .

				'</td><td style="text-align: center;">' .

				'<a href="/results/'. $_ . '_summary_score.html">by score</a>    <a href="/results/' . $_ . '_summary_hits.html">by hits</a><br>' .

				'</td><td style="text-align: center;">' .

				'<a href="/results/'. $_ . '_details_score.html">by score</a>    <a href="/results/' . $_ . '_details_hits.html">by hits</a><br>' .

				"</td></tr>";

	};


	#<a href="' . $uniprot_url . $list_to_print{$n}->{uniprotid} . '" target="_blank">' . $list_to_print{$n}->{uniprotid} . '</a><br>';

	# $link_name = "worksheet" . $_ . "_details.html";

	print SUMMARY "</TABLE>\n";

	print SUMMARY '<TABLE style="width: 500px;" class="boxTable" bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">' . "\n";

	print SUMMARY '<tbody>
```

```
                        <tr>

                        <td style="text-align: center;"><a href="/results/'. $filename
. '.zip" target="_blank">Download the results</a></td>

                    </tr>

                    </tbody>';

      print SUMMARY '</TABLE>';

      print SUMMARY '<br><br><span style="font-weight: bold;"><big>or upload a new file:
</big></span><br><br>';

      print       SUMMARY       '<form       action="../upload.cgi"       method="post"
enctype="multipart/form-data">

                    File to Upload: <input name="excel" type="file">


                          <br>Type of File:   


                      <select name="type">

                      <option value="excel1">Excel File</option>

                      <option value="excel2">Excel File - GI</option>

                      <option value="text1">Text File with GI numbers</option>

                      <option value="text2">Text type 2</option>

                      </select>




                      <input name="Submit" value="Submit File" type="submit">
```

```
                </form>';


        print SUMMARY "<br><h3>This is worksheet #$_, called <i>\"" . $Book->Worksheets($_)->{Name} ."\"</i></h3>";

        print SUMMARY $query->h3("A detailed view of the results sorted by hits follows<br>");

        print SUMMARY '<table style="width: 800px;" class="boxTable" bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">';


        foreach $n (sort {$list_to_print{$b}->{HIT} <=> $list_to_print{$a}->{HIT}} keys %list_to_print){

                print SUMMARY '<tr><th class="right" colspan="2" align="left">gi|' . $list_to_print{$n}->{GI} . ' has ' . $list_to_print{$n}->{HIT} . ' hits and a ' . $list_to_print{$n}->{SCORE} . ' score</th></tr>';

                print SUMMARY '<tr><td bgcolor="#CC9999" width="20%">Protein Name and Organism</td>';

                print SUMMARY '<td width="80%"><i>Name: </i> ' . $list_to_print{$n}->{uniprot_name} . '<br><br>';

                print SUMMARY '<i>Organism: </i>' . $list_to_print{$n}->{uniprot_source} . '<br></td>';


                print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">Keywords</td>';

                print SUMMARY '<td width="80%"><i>Keywords: </i> ' . join(", ", @{$list_to_print{$n}->{keywords}}) . '<br></td>';


                print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">Function</td>';
```

```
            print SUMMARY '<td width="80%"><i>Function: </i> ' .
$list_to_print{$n}->{uniprot_function} . '<br></td>';


            print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">Ontology</td>';

            print SUMMARY '<td width="80%"><i>Ontology: </i> ' . join(", ",
@{$list_to_print{$n}->{functions}}) . '<br></td>';


            print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">Family
Classification</td>';

            print SUMMARY '<td width="80%"><i>InterPro : </i> ' . join(", ",
@{$list_to_print{$n}->{interprodesc}}) . '<br></td>';


            print SUMMARY '<tr><td bgcolor="#CCCCFF" width="20%">External Links</td>';

            print SUMMARY '<td width="80%"><i>NCBI : </i> <a href="' . $ncbi_url .
$list_to_print{$n}->{GI} . '" target="_blank">' . $list_to_print{$n}->{GI} . '</a><br>';

            print SUMMARY '<i>UniProt : </i> <a href="' . $uniprot_url .
$list_to_print{$n}->{uniprotid} . '" target="_blank">' . $list_to_print{$n}->{uniprotid}
. '</a><br>';

            #print SUMMARY '<i>iProClass : </i> <a href="' . $iproclass_url .
$list_to_print{$n}->{ipcid} . '" target="_blank">' . $list_to_print{$n}->{ipcid} .
'</a><br>';

            print SUMMARY '<i>MGI : </i> <a href="' . $mgi_url_start .
$list_to_print{$n}->{accession_number} . $mgi_url_end . '" target="_blank">' .
$list_to_print{$n}->{accession_number} . '</a><br>';

            print SUMMARY '<i>Ensembl : </i> <a href="' . $ensembl_url .
$list_to_print{$n}->{accession_number} . '" target="_blank">' .
$list_to_print{$n}->{accession_number} . '</a><br></td>';


      };

      print SUMMARY '</table>';

      print SUMMARY $query->end_html;
```

```
	close (SUMMARY);
}

sub print_summary{

	my (%list_to_print) = @_;
	if ((keys %list_to_print)==0) { return (0)};  #if the list is empty, which means that the worksheet is empty, do nothing

	my $link_name = $worksheet_names[$_-1]. "_summary_score.html";
	open SUMMARY, ">$results_dir/$link_name";  # or die "Could not create $results_dir/$link_name\n";
	print SUMMARY '<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" xml:lang="en-US"><head><title>' .
				$Book->Worksheets($_)->{Name} .
				' Summary sorted by score</title
				</head><body bgcolor="#CCCCCC"><a href="/"><img style="width: 800px; height: 130px;" alt="banner" src="../protein-banner-illustration.jpg"></a><br>';

	print SUMMARY '<h3>Use this table to navigate through the results</h3>';
	print SUMMARY '<TABLE style="width: 500px;" class="boxTable" bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">' . "\n";
	print SUMMARY "<tr><th>#</th><th>Worksheet</th><th>Summary</th><th>Details</th></tr>\n";
```

```
	$worksheet_number = 0;

		foreach (@worksheet_names)  {

		$worksheet_number+=1;

		print SUMMARY "<tr><td>" .

				$worksheet_number .

				'</td><td style="text-align: center;">' .

				$_  .

				'</td><td style="text-align: center;">' .

				'<a    href="/results/'.    $_    .    '_summary_score.html">by score</a>    <a href="/results/' . $_ . '_summary_hits.html">by hits</a><br>' .

				'</td><td style="text-align: center;">' .

				'<a    href="/results/'.    $_    .    '_details_score.html">by score</a>    <a href="/results/' . $_ . '_details_hits.html">by hits</a><br>' .

				"</td></tr>";

	};


	#<a    href="'    .    $uniprot_url    .    $list_to_print{$n}->{uniprotid}    .    '" target="_blank">' . $list_to_print{$n}->{uniprotid} . '</a><br>';

	# $link_name = "worksheet" . $_ . "_details.html";

	print  SUMMARY "</TABLE>\n";

	print  SUMMARY  '<TABLE  style="width:  500px;"  class="boxTable"  bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">' . "\n";

	print SUMMARY '<tbody>

				<tr>

				<td style="text-align: center;"><a href="/results/'. $filename . '.zip" target="_blank">Download the results</a></td>

			</tr>

			</tbody>';
```

```
print SUMMARY '</TABLE>';

print SUMMARY '<br><br><span style="font-weight: bold;"><big>or upload a new file:
</big></span><br><br>';

print SUMMARY '<form action="../upload.cgi" method="post"
enctype="multipart/form-data">

File to Upload: <input name="excel" type="file">

    <br>Type of File:   

<select name="type">

<option value="excel1">Excel File</option>

<option value="excel2">Excel File - GI</option>

<option value="text1">Text File with GI numbers</option>

<option value="text2">Text type 2</option>

</select>

<input name="Submit" value="Submit File" type="submit">

</form>';
```

```
      print SUMMARY "<br><h3>This is worksheet #$_, called <i>\"" .
$Book->Worksheets($_)->{Name} ."\"</i></h3>";

      print SUMMARY $query->h3("The results sorted by score are:\n");

      print SUMMARY '<TABLE style="width: 800px;" class="boxTable" bgcolor="#ffffff"
border="1" cellpadding="8" cellspacing="0">' . "\n";

    print SUMMARY "<tr><th>Score</th><th>Hits</th><th>Protein</th></tr>\n";


      foreach (sort {$list_to_print{$b}->{SCORE} <=> $list_to_print{$a}->{SCORE}}  keys
%list_to_print)  {

            print SUMMARY      "<tr><td>" .

                                    $list_to_print{$_}->{SCORE}  .

                                    "</td><td>" .

                                    $list_to_print{$_}->{HIT}  .

                                    "</td><td>" .

                                    $list_to_print{$_}->{EXCEL_NAME} .

                                    "</td></tr>";

      };

      print SUMMARY "</TABLE>\n";

      print SUMMARY $query->end_html;

      close (SUMMARY);




      my $link_name = $worksheet_names[$_-1]. "_summary_hits.html";
```

```
	open SUMMARY, ">$results_dir/$link_name"; # or die "Could not create $results_dir/$link_name\n";

	print SUMMARY '<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" xml:lang="en-US"><head><title>' .

				$Book->Worksheets($_)->{Name} .

				' Summary sorted by hits</title

				</head><body bgcolor="#CCCCCC"><a href="/"><img style="width: 800px; height: 130px;" alt="banner" src="../protein-banner-illustration.jpg"></a><br>';



	print SUMMARY '<h3>Use this table to navigate through the results</h3>';

	print SUMMARY '<TABLE style="width: 500px;" class="boxTable" bgcolor="#ffffff" border="1" cellpadding="8" cellspacing="0">' . "\n";

	print SUMMARY "<tr><th>#</th><th>Worksheet</th><th>Summary</th><th>Details</th></tr>\n";

	$worksheet_number = 0;

	foreach (@worksheet_names) {

		$worksheet_number+=1;

		print SUMMARY "<tr><td>" .

				$worksheet_number .

				'</td><td style="text-align: center;">' .

				$_ .

				'</td><td style="text-align: center;">' .

				'<a href="/results/'. $_ . '_summary_score.html">by score</a>    <a href="/results/' . $_ . '_summary_hits.html">by hits</a><br>' .

				'</td><td style="text-align: center;">' .

				'<a href="/results/'. $_ . '_details_score.html">by score</a>    <a href="/results/' . $_ . '_details_hits.html">by hits</a><br>' .
```

```
                        "</td></tr>";
        };

        #<a    href="'    .    $uniprot_url    .    $list_to_print{$n}->{uniprotid}    .    '"
target="_blank">' . $list_to_print{$n}->{uniprotid} . '</a><br>';
        # $link_name = "worksheet" . $_ . "_details.html";
        print  SUMMARY "</TABLE>\n";
        print  SUMMARY  '<TABLE  style="width:  500px;"  class="boxTable"  bgcolor="#ffffff"
border="1" cellpadding="8" cellspacing="0">' . "\n";
        print SUMMARY '<tbody>
                        <tr>
                        <td style="text-align: center;"><a href="/results/'. $filename
. '.zip" target="_blank">Download the results</a></td>
                    </tr>
                    </tbody>';
        print SUMMARY '</TABLE>';
        print SUMMARY '<br><br><span style="font-weight: bold;"><big>or upload a new file:
</big></span><br><br>';
        print       SUMMARY       '<form       action="../upload.cgi"       method="post"
enctype="multipart/form-data">
                    File to Upload: <input name="excel" type="file">

                          <br>Type of File:   

                      <select name="type">
                      <option value="excel1">Excel File</option>
                      <option value="excel2">Excel File - GI</option>
                      <option value="text1">Text File with GI numbers</option>
```

```
                    <option value="text2">Text type 2</option>

                    </select>



                    <input name="Submit" value="Submit File" type="submit">




                </form>';

    print SUMMARY "<br><h3>This is worksheet #$_, called <i>\"" .
$Book->Worksheets($_)->{Name} ."\"</i></h3>";

    print SUMMARY $query->h3("The results  sorted by hits are:\n");

    print SUMMARY '<TABLE style="width: 800px;" class="boxTable" bgcolor="#ffffff"
border="1" cellpadding="8" cellspacing="0">' . "\n";

  print SUMMARY "<tr><th>Score</th><th>Hits</th><th>Protein</th></tr>\n";


    foreach (sort {$list_to_print{$b}->{HIT} <=> $list_to_print{$a}->{HIT}} keys
%list_to_print) {

        print SUMMARY       "<tr><td>" .

                                $list_to_print{$_}->{SCORE}  .

                                "</td><td>" .

                                $list_to_print{$_}->{HIT}  .

                                "</td><td>" .

                                $list_to_print{$_}->{EXCEL_NAME} .

                                "</td></tr>";

    };
```

```
	print SUMMARY "</TABLE>\n";
	print SUMMARY $query->end_html;
	close (SUMMARY);
}

sub standard_header {

	my $file = shift;
	print
	$query->header .
	$query->start_html(-title=>'Results', -bgcolor=>"#CCCCCC") .
	'<a href="/"><img style="width: 800px; height: 130px;" alt="banner"
src="protein-banner-illustration.jpg"></a><br>' .
	$query->h3("Thanks for uploading your results." );
}

sub standard_footer{
	print  $query->end_html;
}

sub table_menu{

	print '<TABLE style="width: 500px;" class="boxTable" bgcolor="#ffffff" border="1"
cellpadding="8" cellspacing="0">' . "\n";
	print "<tr><th>#</th><th>Worksheet</th><th>Summary</th><th>Details</th></tr>\n";

	$worksheet_number=0;
```

```
	foreach (@worksheet_names)  {

		$worksheet_number+=1;

		print  "<tr><td>" .

				$worksheet_number .

				'</td><td style="text-align: center;">' .

				$_  .

				'</td><td style="text-align: center;">' .

				'<a	href="/results/'.	$_	.	'_summary_score.html"
target="_blank">by score</a>    <a href="/results/' . $_ . '_summary_hits.html"
target="_blank">by hits</a><br>' .

				'</td><td style="text-align: center;">' .

				'<a	href="/results/'.	$_	.	'_details_score.html"
target="_blank">by score</a>    <a href="/results/' . $_ . '_details_hits.html"
target="_blank">by hits</a><br>' .

				"</td></tr>";

	};


	#<a	href="'	.	$uniprot_url	.	$list_to_print{$n}->{uniprotid}	.	'"
target="_blank">' . $list_to_print{$n}->{uniprotid} . '</a><br>';

	# $link_name = "worksheet" . $_ . "_details.html";


	print  "</TABLE>\n";


	print '<TABLE style="width: 500px;" class="boxTable" bgcolor="#ffffff" border="1"
cellpadding="8" cellspacing="0">' . "\n";

	print '<tbody>

				<tr>

				<td style="text-align: center;"><a href="/results/'. $filename
. '.zip" target="_blank">Download the results</a></td>
```

```
                </tr>
                </tbody>';
        print '</TABLE>';

}

sub archive_files {
        my $zip = Archive::Zip->new();          #let's zip the files and offer them for
download

        foreach (@worksheet_names){
                $helper1 = 'results/' . $_ . '_details_score.html';
                $helper2 = 'results/' . $_ . '_summary_score.html';
                $helper3 = 'results/' . $_ . '_details_hits.html';
                $helper4 = 'results/' . $_ . '_summary_hits.html';
                $file_member = $zip->addFile( $helper1);
                $file_member = $zip->addFile( $helper2);
                $file_member = $zip->addFile( $helper3);
                $file_member = $zip->addFile( $helper4);
        };

        $helper5 = 'results/' . $filename . '.zip';
        unless ( $zip->writeToFileNamed($helper5) == AZ_OK ) {
                die 'write error';
        }
}
```