ΕΠΛ232 – Προγραμματιστικές Τεχνικές και Εργαλεία

Οργάνωση Προγράμματος & Ανατομία Προγράμματος
(Κεφάλαιο 10, KNK-2ED)

Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου

http://www.cs.ucy.ac.cy/courses/EPL232

Dr. Andreas Aristidou a.aristidou@ieee.org

Περιεχόμενο Διάλεξης 4

- **Οργάνωση Προγράμματος (Κεφ. 10)**
  - *Τοπικές Μεταβλητές* (αυτόματες, static), *Εξωτερικές Μεταβλητές* (block/file scope), Μπλοκ { }, *Εμβέλεια Μεταβλητών*
  - *Οργάνωση Προγράμματος* σε ένα αρχείο κώδικα.
- **(Προκαταρτική) Είσοδος/Εξοδος από Αρχεία**
  - Συναρτήσεις Βιβλιοθήκης fopen, fread, fwrite, παράδειγμα με ψευδοτυχαίους αριθμούς

Dr. Andreas Aristidou a.aristidou@ieee.org

Local Automatic Variables

Τοπικές (Αυτόματες) Μεταβλητές

- Μια **μεταβλητή** που **δηλώνεται στο σώμα** μιας συνάρτησης λέγεται ότι είναι **τοπική (*local*)** στη συνάρτηση:

```
int sum_digits(int n) {
  int sum = 0;   /* local var */
  ...
}
```

```
void f(void)
{                          C99
  ...
  int i;
  ...                      scope of i
}
```

- **Ιδιότητες** τοπικών μεταβλητών (αλλά και παραμέτρων):
  - ***Αυτόματη αποθηκευτική διάρκεια (Automatic storage duration):*** **Δεσμεύεται (allocated)** «αυτόματα» ο χώρος όταν καλείται η συνάρτηση και **αποδεσμεύεται (deallocated)** αυτόματα όταν **ολοκληρωθεί** η εκτέλεση της.
  - ***Εμβέλεια Μπλοκ (Block scope):*** Μια τοπική μεταβλητή είναι **«ορατή»** από τη **δήλωση** της **μέχρι** το **τέλος της συνάρτησης** που την δηλώνει.

Local Static Variables
Τοπικές Στατικές Μεταβλητές

- Προσθέτοντας **`static`** στη δήλωση μιας τοπικής μεταβλητής, καθιστά την μεταβλητή να έχει **στατική αποθηκευτική διάρκεια (*static storage duration.)***
- Δηλαδή, η static μεταβλητή διατηρεί την τιμή της μέχρι την **επόμενη κλήση …**

```
#include <stdio.h>
void foo() {
  static int i;
  printf("%d", i);
  i++;}
```

```
int main() {
  foo(); foo(); foo();
  return 0;}
```

→ Εκτυπώνει:
0 1 2

- **. . .** αλλά η **ορατότητα** της παραμένει σε **επίπεδο block** συνάρτησης, όπως τις **αυτόματες μεταβλητές**
  - Συνεπώς, η μεταβλητή δεν είναι ορατή σε άλλες συναρτήσεις.
- **Χρήση:** Μεταβλητές που θα χρησιμοποιούνται μεταξύ αλλεπάλληλων κλήσεων (για αποφυγή συνεχούς δημιουργίας / καταστροφής τους)
  - **Γενικά συνίσταται να χρησιμοποιείται με μεγάλη ΠΡΟΣΟΧΗ.**

Dr. Andreas Aristidou a.aristidou@ieee.org

Local Static Variables

Τοπικές Στατικές Μεταβλητές

Το `static` μπορεί να χρησιμοποιηθεί τόσο με **πίνακες** όσο και με **δείκτες** που θα δούμε στη συνέχεια

Πως δουλεύει ο ακόλουθος κώδικας;

```
#include <stdio.h> // printf
void foo(int i) {
  static char name[10]="variable";
  printf("%s\n", name);
  name[i]='0';
}
int main() {
  int i = 0;
  foo(i++); foo(i++); foo(i);
  return 0;
}
```

Αποτέλεσμα Προγράμματος:

variable
0ariable
00riable

Μια `global` μεταβλητή δεν χρειάζεται να είναι `static` όπως δείχνει η επόμενη διαφάνεια.

Dr. Andreas Aristidou a.aristidou@ieee.org

External Variables
Εξωτερικές Μεταβλητές

- **Εξωτερικές Μεταβλητές (External Variables):** μεταβλητές που **ΔΕΝ** ορίζονται μέσα στο **σώμα** μιας συνάρτηση και **ΔΕΝ** είναι **παράμετροι** της.
  - Γνωστές ως **καθολικές μεταβλητές (*global variables*)**
- **Ιδιότητες Εξωτερικών Μεταβλητών:**
  - **Στατική Αποθηκευτική Διάρκεια (static storage duration)**
    - Όπως τις στατικές μεταβλητές δεν καταστρέφονται άμεσα
  - **Εμβέλεια Αρχείου (file scope)**
    - Μια μεταβλητή είναι **ορατή** από τη **δήλωση** της μέχρι το **τέλος του αρχείου** που **ορίζει** την **μεταβλητή**

Dr. Andreas Aristidou a.aristidou@ieee.org

External Variables
Εξωτερικές Μεταβλητές

Πλεονεκτήματα & Μειονεκτήματα

- Οι εξωτερικές μεταβλητές είναι χρήσιμες όταν πολλές συναρτήσεις πρέπει να μοιράζονται μια συγκεκριμένη μεταβλητή ή όταν κάποιες συναρτήσεις μοιράζονται ένα μεγάλο αριθμό μεταβλητών.
- Στις περισσότερες περιπτώσεις όμως, είναι καλύτερο για τις συναρτήσεις να επικοινωνούν μέσω παραμέτρων και όχι με την κοινή χρήση μεταβλητών:
  - Εάν αλλάξουμε μια εξωτερική μεταβλητή (π.χ., αλλάζοντας τον τύπο της), θα πρέπει να ελέγξουμε κάθε συνάρτηση στο αρχείο για να δούμε πώς αυτή η αλλαγή την επηρεάζει.
  - Εάν μια εξωτερική μεταβλητή έχει αντιστοιχιστεί σε μια λανθασμένη τιμή, μπορεί να είναι δύσκολο να εντοπίσετε την λανθάνουσα συνάρτηση.
  - Οι συναρτήσεις που βασίζονται σε εξωτερικές μεταβλητές είναι δύσκολο να επαναχρησιμοποιηθούν σε άλλα προγράμματα.

External Variables

Εξωτερικές Μεταβλητές

Πλεονεκτήματα & Μειονεκτήματα

- Μην χρησιμοποιείτε την ίδια εξωτερική μεταβλητή για διαφορετικούς σκοπούς σε διαφορετικές συναρτήσεις.
- Ας υποθέσουμε ότι πολλές συναρτήσεις χρειάζονται μια μεταβλητή που ονομάζεται `i` για τον έλεγχο μιας εντολής `for`.
  - Αντί να δηλώσουμε το `i` σε κάθε συνάρτηση που το χρησιμοποιεί, ορισμένοι προγραμματιστές το δηλώνουν μόνο μία φορά στην αρχή του προγράμματος.
  - Αυτή η πρακτική είναι παραπλανητική. Κάποιος που διαβάζει το πρόγραμμα μπορεί να πιστεύει ότι οι `i` σχετίζονται, αλλά στην πραγματικότητα δεν έχουν κάποια σχέση μεταξύ τους.
- **Με άλλα λόγια:**
  - **Δύσκολο debugging, δύσκολη επαναχρησιμοποίηση κώδικα, σπατάλη μνήμης, για αυτό η χρήση τους να συνίσταται να ΠΕΡΙΟΡΙΣΤΕΙ στο ελάχιστο στο μάθημα**

External Variables
Εξωτερικές Μεταβλητές

- Τι πάει λάθος με το ακόλουθο κώδικα;

```
int i;

void print_one_row(void)
{
  for (i = 1; i <= 10; i++)
    printf("*");
}

void print_all_rows(void)
{                  i = 1
  for (i = 1; i <= 10; i++) {
    print_one_row(); i = 11
    printf("\n");
  }
}
```

Μετά την εκτέλεση του πρώτου `for`, το `i` έχει τιμή 11, αντί 2, οπότε η `for` του `print_all_rows` εκτελείτε μόνο μια φορά

- Η δήλωση εξωτερικών μεταβλητών, όταν αυτές θα έπρεπε να είναι τοπικές, μπορεί να οδηγήσει σε σφάλματα.
- Αυτός ο κώδικας θα έπρεπε να εμφανίζει μια 10 × 10 διάταξη από αστερίσκους:

Αντί να εκτυπώνονται όλες οι γραμμές, εκτυπώνεται μόνο μια! ☹
Γιατί;

Dr. Andreas Aristidou a.aristidou@ieee.org

Blocks

Μπλοκ Εντολών

- Η C, όπως και άλλες γλώσσες, παρέχει την έννοια των **ρητών μπλοκ { }** για να **περιορίσει την εμβέλεια μεταβλητών**.
  - Λιγότερες συγκρούσεις ονομάτων (**name conflicts**)
  - Μείωση συνωστισμού δηλώσεων στην αρχή των συναρτήσεων (**cluttering declarations**)
- Η διάρκεια αποθήκευσης μιας μεταβλητής που δηλώνεται σε ένα μπλοκ είναι αυτόματη: ο χώρος αποθήκευσης για τη μεταβλητή εκχωρείται κατά την εισαγωγή του μπλοκ, και η κατάργηση της κατά την έξοδο
- Η μεταβλητή έχει εμβέλεια μόνο εντός του block. Δεν μπορεί να γίνει αναφορά της έξω από το μπλοκ.
- Μια μεταβλητή που ανήκει σε ένα μπλοκ μπορεί να δηλωθεί ως `static` ώστε να δοθεί η στατική διάρκεια της αποθήκευσης της.

Dr. Andreas Aristidou a.aristidou@ieee.org

Blocks
Μπλοκ Εντολών

```
#include <stdio.h>
void foo(void) {
    // other statements ...
    {
        int i = 5;       // Automatic storage duration, block scope
        static int j; // Static storage duration, block scope
        printf("%d %d\n", i++, j++);
    }
    // printf("%d %d", i, j); - compile error: 'i' 'j' undeclared (first use in this
  function)
}

int main() {
    foo(); foo(); foo();
    return 0;
}
```

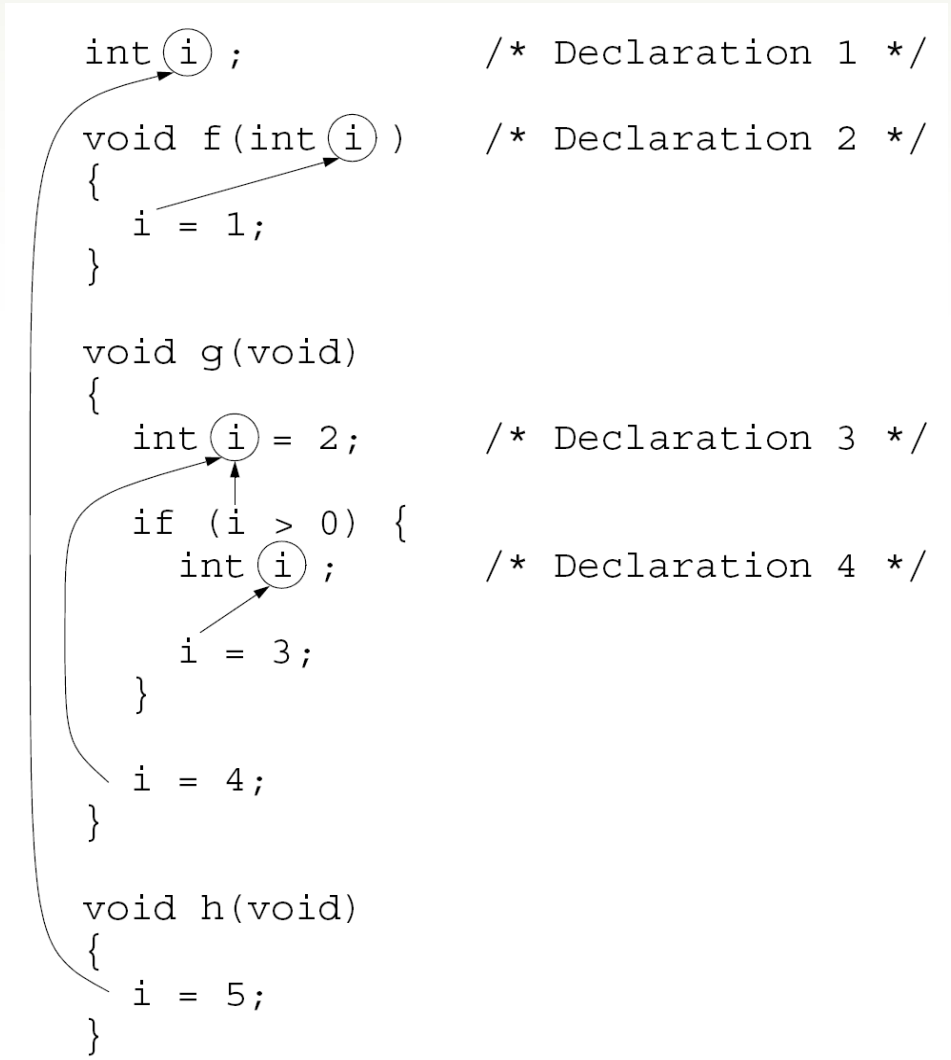
Εκτυπώνει:

5 0
5 1
5 2

Dr. Andreas Aristidou a.aristidou@ieee.org

Variable Scope
Εμβέλεια Μεταβλητών

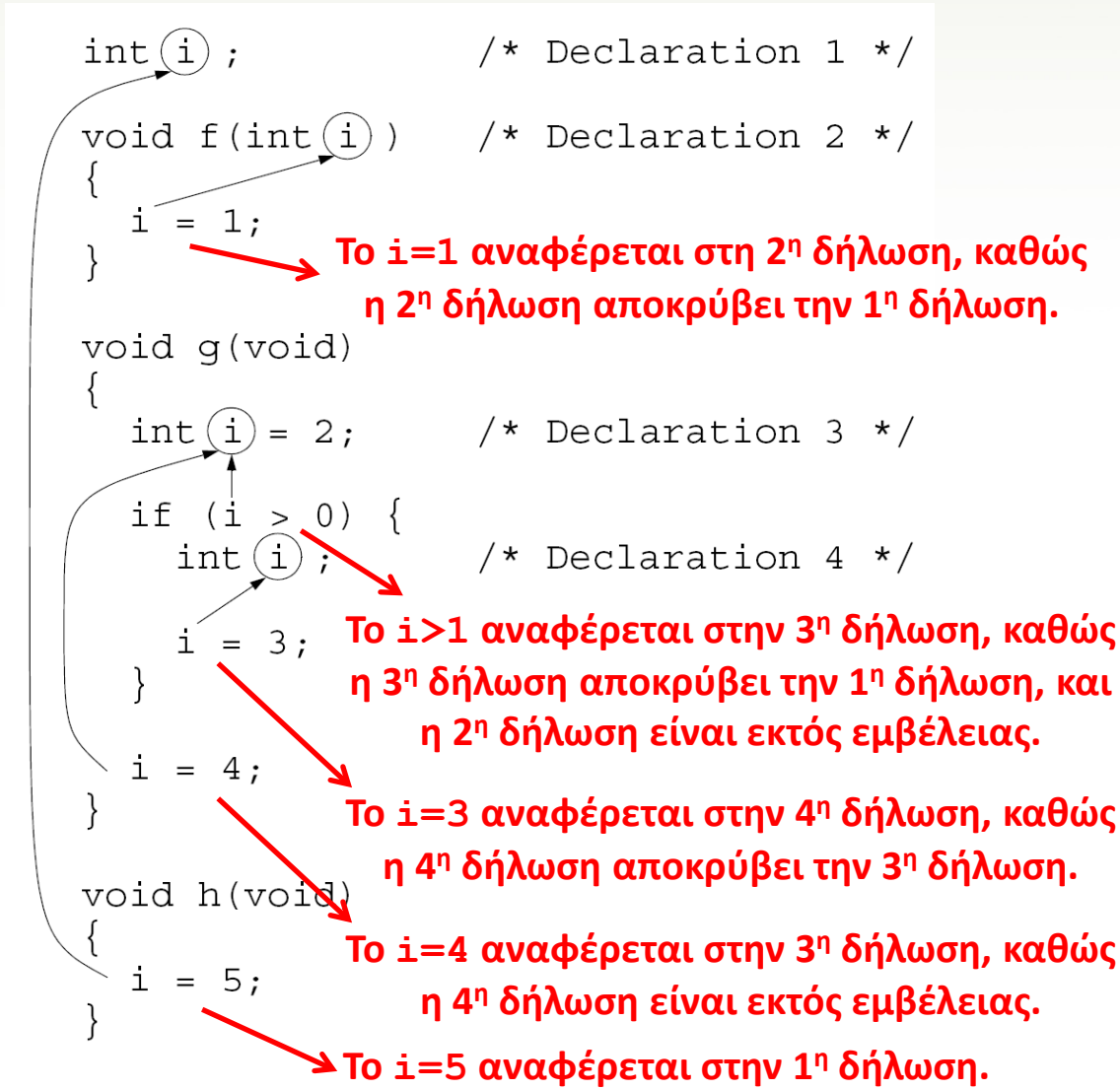
- **Αρχή Εμβέλειας:** Όταν γίνεται **δήλωση** μιας μεταβλητής που είναι **ήδη δηλωμένη** (ορατή), η νέα δήλωση «**αποκρύπτει**» την παλιά **δήλωση**.
  - Στο τέλος η **παλιά δήλωση** αποκτά ξανά το νόημα της.



Dr. Andreas Aristidou a.aristidou@ieee.org

Variable Scope
Εμβέλεια Μεταβλητών

- Σε αυτό το παράδειγμα, η `i` έχει τέσσερις διαφορετικές έννοιες:
  - Στη 1η δήλωση, `i` είναι μια μεταβλητή με διάρκεια στατικής αποθήκευσης και εμβέλεια αρχείου.
  - Στη 2η δήλωση, `i` είναι μια παράμετρος με εμβέλεια μπλοκ.
  - Στην 3η δήλωση, `i` είναι μια αυτόματη μεταβλητή με εμβέλεια μπλοκ.
  - Στην 4η δήλωση, `i` είναι επίσης αυτόματη και έχει εμβέλεια μπλοκ.



Dr. Andreas Aristidou a.aristidou@ieee.org

Οργάνωση Προγράμματος **(με 1 Αρχείο Πηγαίου Κώδικα *)**

- * Αργότερα θα δούμε πως η σημαντική **αρχή της αφαιρετικότητας** μας επιβάλλει να **διασπάσουμε** ένα πρόγραμμα σε **πολλαπλά αρχεία** πηγαίου κώδικα.
- **Βασικά συστατικά** ενός **C προγράμματος** (σε σειρά εμφάνισης):
  - **Οδηγίες προεπεξεργασίας** μεταγλωττιστή (compiler preprocessing directives) such as `#include` and `#define`
  - Ορισμός **Νέων Τύπων** (Type definitions) - `typedef`
  - Δήλωση **Εξωτερικών Μεταβλητών** (Declarations of Ext. vars)
  - **Πρότυπα Συναρτήσεων** (function prototypes)
  - **Ορισμός Συναρτήσεων** (function definitions) - πριν την χρήση τους εναλλακτικά παίρνεται το λάθος `undefined symbol  first referenced in file _print ...`
- Τύποι, μεταβλητές, κτλ., έχουν «ορατότητα» μόνο μετά τον ορισμό τους.

Dr. Andreas Aristidou a.aristidou@ieee.org

Οργάνωση Προγράμματος **(με 1 Αρχείο Πηγαίου Κώδικα *)**

- **Βασικά Συστατικά Σχολίου:**
  - **Name:** Όνομα Συνάρτησης
  - **Description:** Τι ρόλο επιτελεί (Σύντομη και Εκτενέστερη Περιγραφή – Short/Long)
  - **Parameters:** Νόημα κάθε παραμέτρου (`@param`)
  - **Return Value:** Επεξήγηση της τιμής εξόδου, εάν υπαρχει (`@return`)
  - **Others:** Επεξήγηση οποιονδήποτε άλλων σημαντικών δεδομένων και χαρακτηριστικών, π.χ.,
    - περιγραφή των συνεπειών μεταβολής εξωτερικών μεταβλητών
    - αλγόριθμου, δομής δεδομένων, κτλ.

Οργάνωση Προγράμματος **(με 1 Αρχείο Πηγαίου Κώδικα *)**

- Παράδειγμα Σχολίου για Χρήση με **Σύστημα Τεκμηρίωσης Κώδικα (Software Documentation System) Doxygen** (www.doxygen.org/)
  - Έχει μελετηθεί στα εργαστήρια.

Βασικά Συστατικά Σχολίου:

```
/**
 * <A short one line description>
 *
 * <Longer description>
 * <May span multiple lines or paragraphs as needed>
 *
 * @param  Description of method's or function's input parameter
 * @param  ...
 * @return Description of the return value
 */
```

Οργάνωση Προγράμματος (με 1 Αρχείο Πηγαίου Κώδικα *)

```
/**
 * @file
 * @author  John Doe <jdoe@example.com>
 * @version 1.0
 *
 * @section LICENSE
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License as
 * published by the Free Software Foundation; either version 2 of
 * the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful, but
 * WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 * General Public License for more details at
 * http://www.gnu.org/copyleft/gpl.html
 *
 * @section DESCRIPTION
 *
 * The time module represents a moment of time.
 */
```

Παράδειγμα Σχολίων για την Αρχή του Αρχείου

Οργάνωση Προγράμματος **(με 1 Αρχείο Πηγαίου Κώδικα *)**

- **Σκελετός Προγράμματος (Skeleton Program):** Ένα υψηλού επιπέδου πρόγραμμα το όποιο περιέχει **κενό κώδικα (dummy code).**
  - **Χρησιμοποιείται** για να δημιουργηθεί η **βασική δομή** ενός **προγράμματος** και είναι **βασικό εργαλείο** για να **διαμεριστεί η** ανάπτυξη ενός λογισμικού σε μια **ομάδα προγραμματιστών**.
  - Περισσότερα μετά το **midterm** όταν θα μιλήσουμε για **SVN** και αρχές ανάπτυξης **μεγάλων προγραμμάτων**

```
int main(void)
{
  for (;;) {
    read_cards();
    analyze_hand();
    print_result();
  }
}
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

- Το πρόγραμμα `poker.c` ταξινομεί το χαρτί σε μια παρτίδα πόκερ.
- Κάθε χαρτί έχει *suit* και *rank*.
  - Suits: clubs, diamonds, hearts, spades
  - Ranks: two, three, four, five, six, seven, eight, nine, ten, jack, queen, king, ace
- Jokers δεν επιτρέπονται, και οι άσσοι έχουν τον ψηλότερο βαθμό.
- Μετά την ανάγνωση πέντε φύλλων, το πρόγραμμα θα ταξινομήσει το χαρτί χρησιμοποιώντας τις ακόλουθες κατηγορίες.
  - Αν ένα χαρτί ανήκει σε δύο ή περισσότερες κατηγορίες, το πρόγραμμα θα επιλέξει την καλύτερη.

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

- Κατηγορίες (από την καλύτερη στην χειρότερη):
  - straight flush (both a straight and a flush)
  - four-of-a-kind (four cards of the same rank)
  - full house (a three-of-a-kind and a pair)
  - flush (five cards of the same suit)
  - straight (five cards with consecutive ranks)
  - three-of-a-kind (three cards of the same rank)
  - two pairs
  - pair (two cards of the same rank)
  - high card (any other hand)

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

- Για σκοπούς εισόδου, οι ranks και οι suits θα είναι μονά γράμματα:

  Ranks: `2 3 4 5 6 7 8 9 t j q k a`

  Suits: `c d h s` (**Clubs - c, Diamonds – d, Hearts - h, Spades -s**.)

- Ενέργειες που πρέπει να ληφθούν εάν ο χρήστης εισάγει λάθος κάρτα ή προσπαθήσει να εισαγάγει την ίδια κάρτα δύο φορές:
  - Αγνοήστε την κάρτα
  - Έκδοση μηνύματος σφάλματος
  - Ζητήστε άλλη κάρτα
- Η εισαγωγή του αριθμού 0 αντί ενός χαρτιού θα προκαλέσει τερματισμό του προγράμματος.

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

- Ένα πρόγραμμα:

```
Enter a card: 2s
Enter a card: 5s
Enter a card: 4s
Enter a card: 3s
Enter a card: 6s
Straight flush
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
Enter a card: 8c
Enter a card: as
Enter a card: 8c
Duplicate card; ignored.
Enter a card: 7c
Enter a card: ad
Enter a card: 3h
Pair
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
Enter a card: 6s
Enter a card: d2
Bad card; ignored.
Enter a card: 2d
Enter a card: 9c
Enter a card: 4h
Enter a card: ts
High card

Enter a card: 0
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

- Το πρόγραμμα έχει τρεις εργασίες:
  - Να διαβάσει το χαρτί από τα πέντε φύλλα
  - Να αναλύσει το χαρτί σε ζευγάρια, κέντα, και ούτω καθεξής.
  - Να εκτυπώσει την ταξινόμηση του χαρτιού
- Οι συναρτήσεις `read_cards`, `analyze_hand`, και `print_result` θα εκτελέσουν αυτές τις εργασίες.
- Η `main` δεν κάνει τίποτα, απλά καλεί αυτές τις συναρτήσεις μέσα σε ένα βρόγχο.

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

- Οι συναρτήσεις μοιράζονται μια αρκετά μεγάλη ποσότητα πληροφοριών, έτσι θα τις έχουμε να επικοινωνούν μέσω εξωτερικών μεταβλητών.
- Η `read_cards` θα αποθηκεύει πληροφορίες σχετικά με το χαρτί σε διάφορες εξωτερικές μεταβλητές.
- Η `analyze_hand` θα εξετάσει στη συνέχεια αυτές τις μεταβλητές, αποθηκεύοντας τα ευρήματά της σε άλλες εξωτερικές μεταβλητές προς όφελος της `print_result`.

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
/* #include directives go here */

/* #define directives go here */

/* declarations of external variables go here */

/* prototypes */
void read_cards(void);
void analyze_hand(void);
void print_result(void);
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
/**********************************************************
 * main: Calls read_cards, analyze_hand, and print_result *
 *       repeatedly.                                      *
 **********************************************************/
int main(void)
{
  for (;;) {
    read_cards();
    analyze_hand();
    print_result();
  }
}

/**********************************************************
 * read_cards:  Reads the cards into external variables;  *
 *              checks for bad cards and duplicate cards. *
 **********************************************************/
void read_cards(void)
{
  …
}
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
/**********************************************
 * analyze_hand: Determines whether the hand contains a   *
 *               straight, a flush, four-of-a-kind,       *
 *               and/or three-of-a-kind; determines the   *
 *               number of pairs; stores the results into *
 *               external variables.                      *
 **********************************************/
void analyze_hand(void)
{
  …
}

/**********************************************
 * print_result: Notifies the user of the result, using   *
 *               the external variables set by            *
 *               analyze_hand.                            *
 **********************************************/
void print_result(void)
{
  …
}
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

- Πώς πρέπει να αντιπροσωπεύουμε το κάθε χαρτί;
- Η `analyze_hand` θα πρέπει να γνωρίζετε πόσα φύλλα είναι σε κάθε rank και κάθε suit.
- Αυτό υποδηλώνει ότι πρέπει να χρησιμοποιήσουμε δύο πίνακες, τους `num_in_rank` και `num_in_suit`.
  - `num_in_rank[r]` θα είναι ο αριθμός των καρτών με rank `r`.
  - `num_in_suit[s]` θα είναι ο αριθμός των καρτών με suit `s`.
- Θα κωδικοποιήσετε rank ως αριθμούς μεταξύ του 0 και 12.
- Τα suits θα είναι αριθμοί από το 0 ως το 3.

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

- Θα χρειαστούμε επίσης ένα τρίτο πίνακα, `card_exists`, έτσι ώστε η `read_cards` θα εντοπίζει διπλές εμφανίσεις.
- Κάθε φορά που η `read_cards` διαβάζει ένα χαρτί με rank `r` και suit `s`, ελέγχει αν η τιμή των `card_exists[r][s]` είναι αληθής.
  - Σε αυτήν την περίπτωση, η κάρτα είχε εισαχθεί προηγουμένως.
  - Εάν όχι, η `read_cards` εκχωρεί `true` στο `card_exists[r][s]`.

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

- Τόσο η `read_cards`, όσο και η `analyze_hand`, θα χρειαστούν πρόσβαση στους πίνακες `num_in_rank` και `num_in_suit`, έτσι θα πρέπει να είναι εξωτερικές μεταβλητές.
- Ο πίνακας `card_exists` χρησιμοποιείται μόνο από την `read_cards`, έτσι μπορεί να είναι τοπικός σε αυτή την συνάρτηση.
- Κατά κανόνα, οι μεταβλητές πρέπει να γίνονται εξωτερικές μόνο αν είναι απαραίτητο.

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

poker.c

```
/* Classifies a poker hand */

#include <stdbool.h>   /* C99 only */
#include <stdio.h>
#include <stdlib.h>

#define NUM_RANKS 13
#define NUM_SUITS 4
#define NUM_CARDS 5

/* external variables */
int num_in_rank[NUM_RANKS];
int num_in_suit[NUM_SUITS];
bool straight, flush, four, three;
int pairs;   /* can be 0, 1, or 2 */
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
/* prototypes */
void read_cards(void);
void analyze_hand(void);
void print_result(void);

/**********************************************
 * main: Calls read_cards, analyze_hand, and print_result *
 *       repeatedly.                                      *
 **********************************************/
int main(void)
{
  for (;;) {
    read_cards();
    analyze_hand();
    print_result();
  }
}
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
/**********************************************
 * read_cards: Reads the cards into the external *
 *             variables num_in_rank and num_in_suit;     *
 *             checks for bad cards and duplicate cards.  *
 **********************************************/
void read_cards(void)
{
  bool card_exists[NUM_RANKS][NUM_SUITS];
  char ch, rank_ch, suit_ch;
  int rank, suit;
  bool bad_card;
  int cards_read = 0;

  for (rank = 0; rank < NUM_RANKS; rank++) {
    num_in_rank[rank] = 0;
    for (suit = 0; suit < NUM_SUITS; suit++)
      card_exists[rank][suit] = false;
  }

  for (suit = 0; suit < NUM_SUITS; suit++)
    num_in_suit[suit] = 0;
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
while (cards_read < NUM_CARDS) {
   bad_card = false;
   printf("Enter a card: ");
   rank_ch = getchar();
   switch (rank_ch) {
     case '0':             exit(EXIT_SUCCESS);
     case '2':             rank = 0; break;
     case '3':             rank = 1; break;
     case '4':             rank = 2; break;
     case '5':             rank = 3; break;
     case '6':             rank = 4; break;
     case '7':             rank = 5; break;
     case '8':             rank = 6; break;
     case '9':             rank = 7; break;
     case 't': case 'T':   rank = 8; break;
     case 'j': case 'J':   rank = 9; break;
     case 'q': case 'Q':   rank = 10; break;
     case 'k': case 'K':   rank = 11; break;
     case 'a': case 'A':   rank = 12; break;
     default:              bad_card = true;
   }
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
    suit_ch = getchar();
    switch (suit_ch) {
      case 'c': case 'C': suit = 0; break;
      case 'd': case 'D': suit = 1; break;
      case 'h': case 'H': suit = 2; break;
      case 's': case 'S': suit = 3; break;
      default:            bad_card = true;
    }

    while ((ch = getchar()) != '\n')
      if (ch != ' ') bad_card = true;

    if (bad_card)
      printf("Bad card; ignored.\n");
    else if (card_exists[rank][suit])
      printf("Duplicate card; ignored.\n");
    else {
      num_in_rank[rank]++;
      num_in_suit[suit]++;
      card_exists[rank][suit] = true;
      cards_read++;
    }
  }
}
```


Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
/**********************************************
 * analyze_hand: Determines whether the hand contains a   *
 *               straight, a flush, four-of-a-kind,       *
 *               and/or three-of-a-kind; determines the   *
 *               number of pairs; stores the results into *
 *               the external variables straight, flush,  *
 *               four, three, and pairs.                  *
 **********************************************/
void analyze_hand(void)
{
  int num_consec = 0;
  int rank, suit;
  straight = false;
  flush = false;
  four = false;
  three = false;
  pairs = 0;
```


Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
  /* check for flush */
  for (suit = 0; suit < NUM_SUITS; suit++)
    if (num_in_suit[suit] == NUM_CARDS)
      flush = true;

  /* check for straight */
  rank = 0;
  while (num_in_rank[rank] == 0) rank++;
  for (; rank < NUM_RANKS && num_in_rank[rank] > 0; rank++)
    num_consec++;
  if (num_consec == NUM_CARDS) {
    straight = true;
    return;
  }

  /* check for 4-of-a-kind, 3-of-a-kind, and pairs */
  for (rank = 0; rank < NUM_RANKS; rank++) {
    if (num_in_rank[rank] == 4) four = true;
    if (num_in_rank[rank] == 3) three = true;
    if (num_in_rank[rank] == 2) pairs++;
  }
}
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Πρόγραμμα

Ταξινόμηση χαρτιών πόκερ

```
/**********************************************
 * print_result: Prints the classification of the hand,   *
 *               based on the values of the external      *
 *               variables straight, flush, four, three,  *
 *               and pairs.                               *
 **********************************************/
void print_result(void)
{
  if (straight && flush) printf("Straight flush");
  else if (four)         printf("Four of a kind");
  else if (three &&
           pairs == 1)   printf("Full house");
  else if (flush)        printf("Flush");
  else if (straight)     printf("Straight");
  else if (three)        printf("Three of a kind");
  else if (pairs == 2)   printf("Two pairs");
  else if (pairs == 1)   printf("Pair");
  else                   printf("High card");

  printf("\n\n");
}
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Περιεχόμενο Διάλεξης

- **Οργάνωση Προγράμματος (Κεφ. 10)**
  - *Τοπικές Μεταβλητές* (αυτόματες, static), *Εξωτερικές Μεταβλητές* (block/file scope), Μπλοκ { }, *Εμβέλεια Μεταβλητών*
  - *Οργάνωση Προγράμματος* σε ένα αρχείο κώδικα.
- **(Προκαταρτική) Είσοδος/Εξοδος από Αρχεία**
  - Συναρτήσεις Βιβλιοθήκης fopen, fread, fwrite, παράδειγμα με ψευδοτυχαίους αριθμούς

Dr. Andreas Aristidou a.aristidou@ieee.org

Είσοδος/Εξοδος από Αρχεία

Εισαγωγική Επισήμανση

- Για **επεξεργασία αρχείων** θα χρειαστούμε **δείκτες** (*) τους οποίους θα δούμε στις **ερχόμενες 2 διαλέξεις**.
- Για να μπορέσετε **μέχρι τότε** να δουλεύετε με αρχεία, θα σας **υποδειχτεί** τώρα ο **βασικός τρόπος** και η **αναλυτική επεξήγηση** θα γίνει **αργότερα** στην Διάλεξη 9
  - Όπου θα δούμε το File I/O σε βάθος.
    - File I/O: File standard input and output devices handled by C programming language.

Dr. Andreas Aristidou a.aristidou@ieee.org

Είσοδος/Εξοδος από Αρχεία

(Άνοιγμα/Κλείσιμο)

Άνοιγμα/Κλείσιμο Αρχείου fopen (<stdio.h>)

FILE * fopen(char *filename, char *mode);

- Η παράμετρος *filename* υποδεικνύει το όνομα του αρχείου που επιθυμούμε να ανοίξουμε
- Το Mode υποδεικνύει το είδος της πράξης (π.χ. Read, write, read-write, etc)

```
FILE *fp = NULL;
fp = fopen("myfile.txt", "r");
…
fclose(fp);
```

Τύπος Ανοίγματος (εδώ READ)

Όνομα Αρχείου

Οντότητα Διαχείρισης Αρχείου. (file pointer)

Είσοδος/Εξοδος από Αρχεία

(Τρόπος προσπέλασης)

Η παράμετρος mode υποδεικνύει τον τρόπο με τον οποίο θέλουμε να προσπελάσουμε το αρχείο

Mode	Σημασία
r	Μόνο **Ανάγνωση**. Αν το αρχείο δεν υπάρχει, επιστρέφεται NULL
w	Μόνο **Εγγραφή**. Αν το αρχείο δεν υπάρχει δημιουργείται, αν υπάρχει τα περιεχόμενά του καταστρέφονται.
a	**Προσθήκη**. Η προσθήκη γίνετε στο τέλος του αρχείου. Αν το αρχείο δεν υπάρχει, δημιουργείται.
r+	**Ανάγνωση** και **εγγραφή**. Αν το αρχείο δεν υπάρχει, επιστρέφεται NULL.
w+	**Ανάγνωση** και **εγγραφή**. Αν το αρχείο δεν υπάρχει δημιουργείται, αν υπάρχει τα περιεχόμενά του καταστρέφονται.
a+	**Προσθήκη** και **ανάγνωση**. Αν το αρχείο δεν υπάρχει, δημιουργείται.

Είσοδος/Εξοδος από Αρχεία

(Ανάγνωση / Εγγραφή Τιμών)

Το ακόλουθο παράδειγμα δείχνει την χρήση της fopen().

```
#include <stdio.h>
#include <stdlib.h>

int main () {
   FILE * fp;

   fp = fopen ("file.txt", "w+");
   fprintf(fp, "%s %s %s %d", "We", "are", "in", 2019);

   fclose(fp);

   return(0);
}
```

Η εκτέλεση του προγράμματος θα δημιουργήσει ένα αρχείο `file.txt` με το ακόλουθο περιεχόμενο

```
We are in 2019
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Είσοδος/Εξοδος από Αρχεία

(Ανάγνωση / Εγγραφή Τιμών)

Το ακόλουθο παράδειγμα δείχνει την χρήση της fopen().

```
#include <stdio.h>

int main () {
   FILE *fp;
   int c;

   fp = fopen("file.txt","r");
   while(1) {
      c = fgetc(fp);
      if( feof(fp) ) {
         break ;
      }
      printf("%c", c);
   }
   fclose(fp);

   return(0);
}
```

```
We are in 2019
```

Είσοδος/Εξοδος από Αρχεία

(Ανάγνωση / Εγγραφή Τιμών)

```
#include <stdio.h>  // fopen, fscanf, fprintf, fclose, printf
#include <stdlib.h> // EXIT_FAILURE
char filename[]="test.txt";

int main()
{
    FILE *fp = NULL;
    int a;

    if ((fp = fopen(filename, "r")) == NULL)     {
        printf("Error: Unable to open %s\n", filename);
        exit(EXIT_FAILURE);
    }

    // Read an integer from the file
    fscanf(fp, "%d", &a);
    printf("%d",a); // ή fprintf(fp, "Duplicate:%d", a); Με "w" | "a" mode

    // Close the file
    fclose(fp);

    return 0;

}
```

Για χρήση **άλλων τύπων** χρησιμοποιούνται αντίστοιχα ορίσματα με την **printf()**

Είσοδος/Εξοδος από Αρχεία

(Ανάγνωση / Εγγραφή Τιμών)

```
#include <stdio.h>  // fopen, fscanf, fprintf, fclose, printf
#include <stdlib.h> // EXIT_FAILURE
#include <time.h>   // time
char filename[]="test.txt";

int main()
{
   FILE *fp = NULL;
   int i;
   srand(time(NULL));

   if ((fp = fopen(filename, "w")) == NULL)      {
      printf("Error: Unable to open %s\n", filename);
      exit(EXIT_FAILURE);
   }

   for (i=0;i<10;i++) {
      fprintf(fp, "%d\n", rand() % 1000 + 1);
   }

   // Close the file
   fclose(fp);
   return 0;
}
```

Τι μπορεί να κάνει το ακόλουθο πρόγραμμά;

Γιατί χρειάζεται μια συνάρτηση όπως την srand();

End of Line (EOL)
Χαρακτήρας Τερματισμού Γραμμής

- *Το newline character είναι διαφορετικό σε διαφορετικά Λειτουργικά Συστήματα!*
  - **LF (\n):** Multics, Unix and Unix-like systems (Linux, OS X, FreeBSD, AIX, Xenix, etc.), BeOS, Amiga, RISC OS, and other
  - ***CR+LF (\r\n):* Microsoft Windows**, *DOS (MS-DOS, PC DOS, etc.), DEC TOPS-10, RT-11, CP/M, MP/M, Atari TOS, OS/2, Symbian OS, PalmOS, Amstrad CPC, and most other early non-Unix and non-IBM Oses*
  - **CR (\r):** Commodore 8-bit machines, Acorn BBC, ZX Spectrum, TRS-80, Apple II family, Oberon, **Mac OS up to version 9**, MIT Lisp Machine and OS-9
  - 0x9B (Atari 8-bit machines), LF+CR (RISC OS spooled text)
- `dos2unix` *and* `unix2dos` *- DOS/MAC to UNIX text file format converter commands.*

Dr. Andreas Aristidou a.aristidou@ieee.org

End of File (EOF)
Χαρακτήρας Τερματισμού Αρχείου

- Δεν χρησιμοποιείται σε αρχεία αυτός ο χαρακτήρας!
- Το EOF==-1 και υποδηλώνει απλά ότι το αρχείο που διαβάζουμε δεν έχει άλλο χαρακτήρα, έτσι η βιβλιοθήκη stdio.h του glibc επιστρέφει -1

```
while (fscanf(in, "%c", &symbol) != EOF) { …}
```

- Στον πίνακα ASCII υπάρχει ο χαρακτήρας 25 EM (end of medium) ο οποίος χρησιμοποιήθηκε σε αρχεία τύπου Windows στο παρελθόν, αλλά πλέον δεν υπάρχει.
- Αργότερα θα μάθουμε πως με το `hexdump -C file.any` θα μπορούμε να βλέπουμε την byte ακολουθία οποιουδήποτε αρχείου και εκεί θα φανεί ακριβώς τι περιέχεται στο κάθε αρχείο (Άσκηση 4)

Dr. Andreas Aristidou a.aristidou@ieee.org

End of File (EOF)
Χαρακτήρας Τερματισμού Αρχείου

- EOF υποδεικνύει το "end of file". Μια νέα γραμμή (είναι αυτό που συμβαίνει όταν πιέζετε το πλήκτρο ENTER) δεν είναι το τέλος ενός αρχείου, είναι το τέλος μιας γραμμής.
- Η τιμή EOF είναι ίση -1 καθώς πρέπει να είναι διαφορετική από κάθε τιμή που επιστρέφει η getchar και αντιστοιχεί σε χαρακτήρα. Η getchar μετατρέπει οποιαδήποτε τιμή χαρακτήρα unsigned char σε int, που θα είναι συνεπώς μη αρνητική. Δηλώνοντας της μια αρνητική τιμή, τότε διαφέρει!