6ο Φροντιστήριο Δομημένου Προγραμματισμού

Άσκηση 1 Δημιούργησε ένα struct με όνομα Student που περιέχει τα πεδία: name (string), age (int) και grade (float). Διάβασε τα στοιχεία ενός φοιτητή από τον χρήστη και εμφάνισε τα στην οθόνη.

```
#include <stdio.h>
struct Student {
    char name[50];
    int age;
    float grade;
};
int main() {
    struct Student s;
    printf("Enter name: ");
    scanf("%s", s.name);
    printf("Enter age: ");
    scanf("%d", &s.age);
    printf("Enter grade: ");
    scanf("%f", &s.grade);
    printf("\nStudent Information:\n");
    printf("Name: %s\n", s.name);
    printf("Age: %d\n", s.age);
    printf("Grade: %.2f\n", s.grade);
    return 0;
}
```

Επεξήγηση

- Δημιουργούμε ένα struct που ονομάζεται Student.
- Τα πεδία είναι name, age και grade.
- Στο main, δηλώνουμε μια μεταβλητή s τύπου Student.
- Χρησιμοποιούμε scanf για να διαβάσουμε τις τιμές από τον χρήστη.
- Τέλος, εμφανίζουμε τα στοιχεία με printf.

Άσκηση 2 Δημιούργησε πίνακα 5 φοιτητών και υπολόγισε τον μέσο όρο των βαθμών τους.

```
#include <stdio.h>
struct Student {
    char name[50];
    int age;
    float grade;
};
int main() {
    struct Student students[5];
    float sum = 0;

    for(int i = 0; i < 5; i++) {
        printf("Enter name of student %d: ", i+1);
        scanf("%s", students[i].name);
        printf("Enter age: ");
        scanf("%d", &students[i].age);
        printf("Enter grade: ");
        scanf("%f", &students[i].grade);

        sum += students[i].grade;
    }
    printf("\nAverage grade: %.2f\n", sum / 5);
    return 0;
}
```

Επεξήγηση

- Δημιουργούμε πίνακα students 5 στοιχείων τύπου Student.
- Διαβάζουμε τα στοιχεία κάθε φοιτητή με ένα βρόχο for.
- Αθροίζουμε τους βαθμούς σε sum και υπολογίζουμε τον μέσο όρο.

Άσκηση 3 Δημιούργησε struct Point με πεδία x και y. Γράψε συνάρτηση που υπολογίζει την απόσταση δύο σημείων.

```
#include <stdio.h>
#include <math.h>
struct Point {
    float x;
    float y;
};
float distance(struct Point p1, struct Point p2) {
    return sqrt((p2.x - p1.x)*(p2.x - p1.x) + (p2.y -
p1.y)*(p2.y - p1.y));
}
int main() {
    struct Point a, b;
    printf("Enter x and y for point A: ");
    scanf("%f %f", &a.x, &a.y);
    printf("Enter x and y for point B: ");
    scanf("%f %f", &b.x, &b.y);
    printf("Distance between A and B: %.2f\n",
distance(a, b));
    return 0;
}
```

Επεξήγηση

- Το struct Point περιέχει τις συντεταγμένες x και y.
- Η συνάρτηση distance υπολογίζει την Ευκλείδεια απόσταση.
- Στο main, διαβάζουμε δύο σημεία και εμφανίζουμε την απόσταση τους.

Άσκηση 4 Δημιούργησε struct Book με πεδία title, author, year. Γράψε συνάρτηση printBook που εμφανίζει τα στοιχεία ενός βιβλίου.

```
#include <stdio.h>
struct Book {
    char title[100];
    char author[50];
    int year;
};
void printBook(struct Book b) {
    printf("Title: %s\n", b.title);
    printf("Author: %s\n", b.author);
    printf("Year: %d\n", b.year);
}
int main() {
    struct Book book;
    printf("Enter book title: ");
    scanf("%s", book.title);
    printf("Enter author: ");
    scanf("%s", book.author);
    printf("Enter year: ");
    scanf("%d", &book.year);
    printf("\nBook Details:\n");
    printBook(book);

    return 0;
}
```

Επεξήγηση

- Δημιουργούμε struct Book.
- Η συνάρτηση printBook δέχεται ένα struct Book και εμφανίζει τα πεδία του.
- Χρησιμοποιούμε scanf για να διαβάσουμε τα στοιχεία του βιβλίου.

Άσκηση 5 Δημιούργησε struct Date (day, month, year) και struct Employee με name, salary και hireDate τύπου Date. Διάβασε και εμφάνισε τα στοιχεία ενός εργαζομένου.

```
#include <stdio.h>
struct Date {
    int day;
    int month;
    int year;
};
struct Employee {
    char name[50];
    float salary;
    struct Date hireDate;
};
int main() {
    struct Employee emp;
    printf("Enter employee name: ");
    scanf("%s", emp.name);
    printf("Enter salary: ");
    scanf("%f", &emp.salary);
    printf("Enter hire date (day month year): ");
scanf("%d %d %d", &emp.hireDate.day, &emp.hireDate.month, &emp.hireDate.year);
    printf("\nEmployee Details:\n");
    printf("Name: %s\n", emp.name);
    printf("Salary: %.2f\n", emp.salary);
    printf("Hire Date: %02d/%02d/%04d\n",
emp.hireDate.day, emp.hireDate.month,
emp.hireDate.year);
    return 0;
}
```

Επεξήγηση

- Το struct Employee περιέχει ένα nested struct Date.
- Δείχνει πώς μπορούμε να ομαδοποιούμε πολύπλοκα δεδομένα.
- Η πρόσβαση στα nested πεδία γίνεται με emp.hireDate.day, emp.hireDate.month, κλπ.

Άσκηση 6 Δημιούργησε πίνακα 5 φοιτητών (Student με name και grade) και ταξινόμησέ τους κατά βαθμό (από υψηλότερο σε χαμηλότερο).

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

struct Student {
    char name[50];
    float grade;
};

int main() {
    struct Student students[5], temp;

    // Εισαγωγή δεδομένων
    for(int i=0; i<5; i++){
        printf("Enter name of student %d: ", i+1);
        scanf("%s", students[i].name);
        printf("Enter grade: ");
        scanf("%f", &students[i].grade);
    }
    // Ταξινόμηση με απλή μέθοδο bubble sort
    for(int i=0; i<4; i++){
        for(int j=i+1; j<5; j++){
            if(students[i].grade < students[j].grade){
                temp = students[i];
                students[i] = students[j];
                students[j] = temp;
            }
        }
    }
    // Εμφάνιση αποτελεσμάτων
    printf("\nStudents sorted by grade:\n");
    for(int i=0; i<5; i++){
        printf("%s: %.2f\n", students[i].name, students[i].grade);
    }
    return 0;
}
```

Επεξήγηση

- Χρησιμοποιούμε πίνακα struct students[5].
- Η ταξινόμηση γίνεται με bubble sort συγκρίνοντας τα grade.
- Ανταλλάσσουμε structs με temp.
- Εκτυπώνουμε τα αποτελέσματα μετά την ταξινόμηση.

Άσκηση 7 Δημιούργησε struct Student και χρησιμοποίησε δείκτη για να αλλάξεις τον βαθμό ενός φοιτητή.

```
#include <stdio.h>
struct Student {
    char name[50];
    float grade;
};
int main() {
    struct Student s;
    struct Student *ptr = &s;
    printf("Enter student name: ");
    scanf("%s", ptr->name);
    printf("Enter grade: ");
    scanf("%f", &ptr->grade);
    printf("\nBefore update: %s - %.2f\n", ptr->name, ptr->grade);
    // Αλλαγή βαθμού μέσω δείκτη
    ptr->grade += 5;
    printf("After update: %s - %.2f\n", ptr->name, ptr->grade);
    return 0;
}
```

Επεξήγηση

- Δημιουργούμε δείκτη ptr που δείχνει στο struct s.
- Με το -> μπορούμε να προσπελάσουμε τα πεδία.
- Αλλάζουμε το πεδίο grade μέσω του δείκτη.

Άσκηση 8 Χρησιμοποίησε malloc για να δημιουργήσεις δυναμικά έναν πίνακα n φοιτητών και υπολόγισε τον μέσο όρο βαθμών.

```
#include <stdio.h>
#include <stdlib.h>
struct Student {
    char name[50];
    float grade;
};
int main() {
    int n;
    printf("Enter number of students: ");
    scanf("%d", &n);
    struct Student *students = (struct Student*)
           malloc(n * sizeof(struct Student));
    float sum = 0;
    for(int i=0; i<n; i++){
        printf("Enter name of student %d: ", i+1);
        scanf("%s", students[i].name);
        printf("Enter grade: ");
        scanf("%f", &students[i].grade);
        sum += students[i].grade;
    }
    printf("\nAverage grade: %.2f\n", sum/n);
    free(students); // Απελευθέρωση μνήμης
    return 0;
}
```

Επεξήγηση

- Χρησιμοποιούμε malloc για δυναμική κατανομή πίνακα structs.
- Η πρόσβαση γίνεται κανονικά με students[i].field.
- Μετά τη χρήση απελευθερώνουμε τη μνήμη με free().

Άσκηση 9 Δημιούργησε struct Team με name και πίνακα 3 παικτών. Διάβασε τα ονόματα των παικτών και εμφάνισέ τα.

```
#include <stdio.h>

struct Team {
    char name[50];
    char players[3][50];
};

int main() {
    struct Team t;

    printf("Enter team name: ");
    scanf("%s", t.name);

    for(int i=0; i<3; i++){
        printf("Enter player %d name: ", i+1);
        scanf("%s", t.players[i]);
    }

    printf("\nTeam %s:\n", t.name);
    for(int i=0; i<3; i++){
        printf("Player %d: %s\n", i+1,
t.players[i]);
    }

    return 0;
}
```

Επεξήγηση

- Έχουμε πίνακα από strings players[3][50] μέσα στο struct.
- Η πρόσβαση γίνεται με t.players[i].

Άσκηση 10 Δημιούργησε πίνακα Student και βρες τον φοιτητή με τον μεγαλύτερο βαθμό.

```
#include <stdio.h>

struct Student {
    char name[50];
    float grade;
};
int main() {
    struct Student students[5];
    for(int i=0; i<5; i++){
        printf("Enter name of student %d: ", i+1);
        scanf("%s", students[i].name);
        printf("Enter grade: ");
        scanf("%f", &students[i].grade);
    }
    int maxIndex = 0;
    for(int i=1; i<5; i++){
        if(students[i].grade > students[maxIndex].grade){
            maxIndex = i;
        }
    }
    printf("\nTop student: %s with grade %.2f\n", students[maxIndex].name, students[maxIndex].grade);
    return 0;
}
```

Επεξήγηση

- Αρχικοποιούμε maxIndex = 0.
- Στο βρόχο συγκρίνουμε κάθε βαθμό με τον μέγιστο μέχρι τώρα.
- Εμφανίζουμε τον φοιτητή με τον μεγαλύτερο βαθμό.