6η Φροντιστηριακή Άσκηση στη C++: Δομές & Ενώσεις

Άσκηση 1: Δημιουργία και Χρήση Δομής

Εκφώνηση:
Γράψτε ένα πρόγραμμα που δηλώνει μια δομή `Student`, η οποία περιέχει όνομα, ηλικία και βαθμό. Στη συνέχεια, δημιουργήστε ένα αντικείμενο αυτής της δομής και εμφανίστε τα στοιχεία του.

Λύση:

```
#include <iostream>
using namespace std;

struct Student {
    string name;
    int age;
    float grade;
};

int main() {
    Student s1 = {"Αλέξανδρος", 20, 8.5};
    cout << "Όνομα: " << s1.name << "\nΗλικία: " << s1.age <<
"\nΒαθμός: " << s1.grade << endl;
    return 0;
}
```

Άσκηση 2: Πίνακας από Δομές

Εκφώνηση:
Γράψτε ένα πρόγραμμα που χρησιμοποιεί έναν πίνακα από δομές `Student` για την αποθήκευση δεδομένων τριών φοιτητών και εμφανίζει τα στοιχεία τους.

Λύση:

```
#include <iostream>
using namespace std;

struct Student {
    string name;
    int age;
    float grade;
};

int main() {
    Student students[3] = { {"Αννα", 19, 9.0}, {"Γιώργος", 21, 7.5},
{"Μαρία", 22, 8.0} };
    for (int i = 0; i < 3; i++) {
        cout << "Όνομα: " << students[i].name << "\nΗλικία: " <<
students[i].age << "\nΒαθμός: " << students[i].grade << "\n" << endl;
    }
    return 0;
}
```

Άσκηση 3: Χρήση Ένωσης

Εκφώνηση:
Γράψτε ένα πρόγραμμα που δηλώνει μια ένωση `Data`, η οποία μπορεί να αποθηκεύσει είτε έναν ακέραιο είτε έναν δεκαδικό αριθμό. Δοκιμάστε την αποθήκευση και των δύο τύπων δεδομένων.

Λύση:

```
#include <iostream>
using namespace std;

union Data {
    int intValue;
    float floatValue;
};

int main() {
    Data d;
    d.intValue = 42;
    cout << "Ακέραιος: " << d.intValue << endl;
    d.floatValue = 3.14;
    cout << "Δεκαδικός: " << d.floatValue << endl;
    return 0;
}
```

Άσκηση 4: Χρήση Δομών μέσα σε Δομές

Εκφώνηση:
Γράψτε ένα πρόγραμμα που χρησιμοποιεί μια δομή `Address` μέσα σε μια δομή `Person` για την αποθήκευση πληροφοριών ενός ατόμου.

Λύση:

```
#include <iostream>
using namespace std;

struct Address {
    string city;
    string street;
};

struct Person {
    string name;
    int age;
    Address address;
};

int main() {
    Person p = {"Κώστας", 30, {"Αθήνα", "Πατησίων 10"}};
    cout << "Όνομα: " << p.name << "\nΗλικία: " << p.age << "\nΠόλη:
" << p.address.city << "\nΟδός: " << p.address.street << endl;
    return 0;
}
```

Άσκηση 5: Προσομοίωση Register

Εκφώνηση:
Χρησιμοποιείστε ένα union για να αναπαραστήσετε έναν καταχωρητή (register) 8-bit, επιτρέποντας πρόσβαση τόσο σε μεμονωμένα bits όσο και σε ολόκληρο το byte.

```
#include <iostream>

// Ορίζουμε ένα union για να αναπαραστήσουμε ένα καταχωρητή
(register)

union StatusRegister {

    struct {

        unsigned char bit0 : 1;  // 1 bit για το bit 0

        unsigned char bit1 : 1;  // 1 bit για το bit 1

        unsigned char bit2 : 1;  // 1 bit για το bit 2

        unsigned char reserved : 5;  // 5 bits που δεν χρησιμοποιούνται

    };

    unsigned char value;  // Το πλήρες byte του καταχωρητή

};



int main() {

    StatusRegister reg;  // Δημιουργούμε ένα αντικείμενο του StatusRegister

    reg.value = 0b00000101;  // Ορίζουμε το 1ο και 3ο bit (bit 0 και bit 2)

    // Εκτυπώνουμε την κατάσταση των bits

    std::cout << "Bit 0: " << static_cast<int>(reg.bit0) << std::endl;

    std::cout << "Bit 1: " << static_cast<int>(reg.bit1) << std::endl;

    std::cout << "Bit 2: " << static_cast<int>(reg.bit2) << std::endl;

    std::cout << "reserved: " << static_cast<int>(reg.reserved) << std::endl;

    std::cout << "value: " << static_cast<int>(reg.value) << std::endl;

    return 0;

}
```