Egzamin (drugi termin 2023) - Programowanie strukturalne - Zestaw M24

Zadanie 1: 6 pkt. Zadanie 2: 12 pkt. Zadanie 3: 14 pkt. Zadanie 4: 18 pkt.

Punktacja: 46-50 pkt - bdb(5,0); 41-45 pkt - db+(4,5); 36-40 pkt - db(4,0); 31-35 pkt - dst+(3,5); 26-30 pkt - dst(3,0); 0-25 pkt - ndst (2,0).

Rozwiązania mają być umieszczone zgodnie ze specyfikacją:

- Zadania powinny być umieszczone w archiwum .zip na udostępnionym pendrive.
- Nazwa archiwum powinna być wg schematu NUMERZESTAWU_NUMERALBUMU.zip gdzie numer zestawu znajduje się na górze kartki z poleceniami. np. A23_123456.zip
- We wnętrzu archiwum powinny znajdować się tylko same kody w języku C, pliki powinny posiadać dokładnie nazwy (z uwzględnieniem wielkości znaków): `zad1.c`, `zad2.c`, `zad3.c`, `zad4.c`.
- Maksymalna waga archiwum 10 MB.
- Archiwum powinno być bez hasła.
- W przypadku pominięcie danego zadania, należy dodać plik o nazwie sprecyzowanej wyżej (zawartość może być pusta).
- Kod zakomentowany nie będzie sprawdzany.

Za zachowanie specyfikacji dokładnie otrzymuję się dodatkowe 2 punkty. Zadania znacznie odbiegające od specyfikacji mogą nie być sprawdzane.

Polecenia są na odwrocie.

1. W folderze DebugXYZ (XYZ - losowowe znaki) znajduje się projekt z kodem w języku C. W pliku main.c w niektórych linijkach są komentarze. Twoim zadaniem jest wpisanie wartości odpowiednich zmiennych po wykonaniu konkretnej linii kodu. Dopisanie nowych linijek czy zaburzenie struktury kodu oznacza zero punktów za polecenie. W przypadku znaków, należy zapisać sam znak w apostrofach np. `'c'` (wielkość znaków ma znaczenie).
2. Napisz funkcję `compare_pointed_numbers`, która ma dwa argumenty. Pierwszym argumentem jest wskaźnik `num1` na stałą wartość typu `float`, a drugim argumentem jest stały wskaźnik `num2` na zmienną typu `float`. Funkcja `compare_pointed_numbers` ma zwrócić 0 jeśli wartości są równe, 1 jeśli pierwsza jest większa i -1 jeśli druga jest większa. Stwórz przypadek testowy dla funkcji.
3. Napisz funkcję, której argumentem jest dwuwymiarowa kwadratowa tablica tablic (zawierająca elementy typu `int`) i jej wymiar $n, n > 0$. Funkcja ma sumę indeksów najmniejszego elementu w tablicy. W przypadku kilku najmniejszych elementów, ma być to największa możliwa suma. Stwórz przypadek testowy.
4. Napisz funkcję, która otrzymuje jako argument listę bez głowy o elementach typu:

```
struct elementList {
    int a;
    struct elementList * next;
};
```

Funkcja ma wyświetlić wartość ostatniego elementu ujemnego. W przypadku pustej listy lub gdy lista nie ma elementów ujemnych - funkcja ma nic nie robić. Stwórz przypadek testowy.