Egzamin (pierwszy termin 2023) - Programowanie strukturalne - Zestaw E47

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.

Zad.1. Dane są następujące wyrazy i znaki:

```
char int int suma tab a a [ ] ( ) , [ ]
```

Ułóż je we właściwej kolejności (zachowując podaną krotność), aby otrzymać nagłówek funkcji `suma`, która dostaje jako argumenty kolejno tablicę elementów i liczbę całkowitą. Następnie dodaj dowolną implementację funkcji i stwórz dla niej przypadek testowy.

Zad.2. Napisz funkcję `copy_squared_value_protected()`, która przyjmuje dwa argumenty: wskaźnik na stałą typu `int` oraz wskaźnik na zmienną typu `int`. Funkcja powinna przepisać wartość stałej podniesionej do kwadratu do zmiennej wskazywanej przez drugi argument, ale tylko jeśli wartość drugiej zmiennej jest równa 0. W przeciwnym przypadku funkcja nie powinna nic zmieniać. Stwórz przypadek testowy, w którym wywołasz funkcję `copy_squared_value_protected()` dla różnych wartości zmiennej i sprawdzisz, czy wartość została poprawnie przepisana tylko wtedy, gdy zmienna miała wartość początkową równą 0.

Zad.3. Utwórz typ wyliczeniowy `Day` reprezentujący dni tygodnia. Napisz funkcję `print_days()`, która przyjmuje jako argumenty wartość typu `Day` i liczbę `n`. Funkcja powinna wydrukować bieżący dzień tygodnia, a następnie rekurencyjnie wywołać siebie z następnym dniem tygodnia, dekrementując `n` przy każdym wywołaniu, aż `n` spadnie do 0. Stwórz przypadek testowy dla funkcji.

Zad.4. Napisz funkcję, która otrzymuje jako argument listę z głową o elementach typu:

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

Funkcja powinna zastąpić wartość każdego zerowego elementu na liście wartością największego elementu (o ile lista ma co najmniej jeden element). Stwórz przypadek testowy.

Przykład: `3,4,-5,0,3,6 -> 3,4,-5,6,3,6`