50 unikalnych i konkretnych funkcji w jezyku C

Funkcja: Zamien wartosci 2 zmiennych

```
void zamien(int *a, int *b) {
    int tmp = *a;
    *a = *b;
    *b = tmp;
}
```

Funkcja: Suma elementow tablicy

```
int suma(int n, const int tab[]) {
    int s = 0;
    for (int i = 0; i < n; i++) s += tab[i];
    return s;
}
```

Funkcja: Srednia arytmetyczna z tablicy

```
float srednia(int n, const int tab[]) {
    int s = 0;
    for (int i = 0; i < n; i++) s += tab[i];
    return (float)s / n;
}
```

Funkcja: Potega n^m

```
int potega(int n, int m) {
    int wynik = 1;
    for (int i = 0; i < m; i++) wynik *= n;
    return wynik;
}
```

Funkcja: Maksimum z dwoch liczb

```
int maksimum(int a, int b) {
    return (a > b) ? a : b;
}
```

Funkcja: Minimum z dwoch liczb

```
int minimum(int a, int b) {
    return (a < b) ? a : b;
}
```

Funkcja: Wartosc bezwzgledna

```
int abs(int x) {
    return (x < 0) ? -x : x;
}
```

Funkcja: Czy liczba parzysta

```
int czy_parzysta(int x) {
    return x % 2 == 0;
}
```

Funkcja: Wyzeruj tablice

```
void wyzeruj(int n, int tab[]) {
    for (int i = 0; i < n; i++) tab[i] = 0;
}
```

Funkcja: Przypisz indeksy do tablicy

```
void wpisz_indeksy(int n, int tab[]) {
    for (int i = 0; i < n; i++) tab[i] = i;
}
```

Funkcja: Wartosc bezwzgledna dla tablicy

```
void bezwzgledne(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        if (tab[i] < 0) tab[i] = -tab[i];
}
```

Funkcja: Sprawdz czy liczba jest pierwsza

```
int czy_pierwsza(int n) {
    if (n < 2) return 0;
    for (int i = 2; i * i <= n; i++)
        if (n % i == 0) return 0;
    return 1;
}
```

Funkcja: Silnia (rekurencyjna)

```
int silnia(int n) {
    if (n <= 1) return 1;
    return n * silnia(n - 1);
}
```

Funkcja: Silnia (iteracyjna)

```
int silnia_it(int n) {
    int wynik = 1;
    for (int i = 2; i <= n; i++) wynik *= i;
    return wynik;
}
```

Funkcja: Ciag Fibonacciego

```
int fib(int n) {
    if (n == 0) return 0;
    if (n == 1) return 1;
    return fib(n - 1) + fib(n - 2);
}
```

Funkcja: Wypisz liczby od n do 1 (rekurencja)

```
void wypisz_wstecz(int n) {
    if (n > 0) {
        printf("%d ", n);
        wypisz_wstecz(n - 1);
    }
}
```

Funkcja: Wypisz liczby od 1 do n (rekurencja)

```
void wypisz_od1(int n) {
    if (n > 0) {
        wypisz_od1(n - 1);
        printf("%d ", n);
    }
}
```

Funkcja: Zamien znaki tablicy na dodatnie

```
void bezwzgledna_char(int n, char tab[]) {
    for (int i = 0; i < n; i++)
        if (tab[i] < 0) tab[i] = -tab[i];
}
```

Funkcja: Wypelnij tablice liczba k

```
void wypelnij(int n, int tab[], int k) {
    for (int i = 0; i < n; i++) tab[i] = k;
}
```

Funkcja: Znajdz maksimum w tablicy

```
int max_tab(int n, int tab[]) {
    int max = tab[0];
    for (int i = 1; i < n; i++)
        if (tab[i] > max) max = tab[i];
    return max;
}
```

Funkcja: Odwroc tablice

```
void odwroc(int n, int tab[]) {
    for (int i = 0; i < n / 2; i++) {
        int tmp = tab[i];
        tab[i] = tab[n - 1 - i];
        tab[n - 1 - i] = tmp;
    }
}
```

Funkcja: Podwój liczby wieksze od sredniej

```
void podwoj_pow_sredniej(int n, int tab[]) {
    float sr = srednia(n, tab);
    for (int i = 0; i < n; i++)
        if (tab[i] > sr) tab[i] *= 2;
}
```

Funkcja: Zlicz liczby podzielne przez 3

```
int zlicz_3(int n, int tab[]) {
    int c = 0;
    for (int i = 0; i < n; i++)
        if (tab[i] % 3 == 0) c++;
    return c;
}
```

Funkcja: Przesun tablice w prawo

```
void przesun_prawo(int n, int tab[]) {
    int last = tab[n - 1];
    for (int i = n - 1; i > 0; i--)
        tab[i] = tab[i - 1];
    tab[0] = last;
}
```

Funkcja: Przesun tablice w lewo

```
void przesun_lewo(int n, int tab[]) {
    int first = tab[0];
    for (int i = 0; i < n - 1; i++)
        tab[i] = tab[i + 1];
    tab[n - 1] = first;
}
```

Funkcja: Zlicz ile liczb jest wiekszych od 10

```
int wieksze_od_10(int n, int tab[]) {
    int c = 0;
    for (int i = 0; i < n; i++)
        if (tab[i] > 10) c++;
    return c;
}
```

Funkcja: Ustaw na 0 wszystkie elementy wieksze od 50

```
void zeruj_50(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        if (tab[i] > 50) tab[i] = 0;
}
```

Funkcja: Usun liczby ujemne z tablicy (ustaw na 0)

```
void usun_ujemne(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        if (tab[i] < 0) tab[i] = 0;
}
```

Funkcja: Zmien elementy na parzyste (dodaj 1 do nieparzystych)

```
void wymus_parzyste(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        if (tab[i] % 2 != 0) tab[i]++;
}
```

Funkcja: Zmien znaki w tablicy

```
void zmien_znaki(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        tab[i] = -tab[i];
}
```

Funkcja: Zastap kazdy element jego reszta z dzielenia przez 5

```
void mod5(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        tab[i] %= 5;
}
```

Funkcja: Mnoz nieparzyste elementy przez 3

```
void mnoz_nieparzyste(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        if (tab[i] % 2 != 0) tab[i] *= 3;
}
```

Funkcja: Ustaw 1 dla dodatnich, -1 dla ujemnych, 0 dla zera

```
void znakowa(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        tab[i] = (tab[i] > 0) ? 1 : (tab[i] < 0) ? -1 : 0;
}
```

Funkcja: Zlicz zera w tablicy

```
int zlicz_zera(int n, int tab[]) {
    int c = 0;
    for (int i = 0; i < n; i++)
        if (tab[i] == 0) c++;
    return c;
}
```

Funkcja: Zlicz liczby mniejsze od podanej wartosci

```
int mniejsze_od(int n, int tab[], int granica) {
    int c = 0;
    for (int i = 0; i < n; i++)
        if (tab[i] < granica) c++;
    return c;
}
```

Funkcja: Zlicz liczby rowne sredniej

```
int rowne_sredniej(int n, const int tab[]) {
    float sr = srednia(n, tab);
    int c = 0;
    for (int i = 0; i < n; i++)
        if (tab[i] == (int)sr) c++;
    return c;
}
```

Funkcja: Obroc tablice o jeden element w lewo

```
void obroc_lewo(int n, int tab[]) {
    int tmp = tab[0];
    for (int i = 0; i < n - 1; i++) tab[i] = tab[i + 1];
    tab[n - 1] = tmp;
}
```

Funkcja: Obroc tablice o jeden element w prawo

```
void obroc_prawo(int n, int tab[]) {
    int tmp = tab[n - 1];
    for (int i = n - 1; i > 0; i--) tab[i] = tab[i - 1];
    tab[0] = tmp;
}
```

Funkcja: Policz ile liczb to kwadraty liczb calkowitych

```
int ile_kwadratow(int n, const int tab[]) {
    int c = 0;
    for (int i = 0; i < n; i++) {
        int j = 0;
        while (j * j < tab[i]) j++;
        if (j * j == tab[i]) c++;
    }
    return c;
}
```

Funkcja: Czy wszystkie liczby sa dodatnie

```
int wszystkie_dodatnie(int n, const int tab[]) {
    for (int i = 0; i < n; i++)
        if (tab[i] <= 0) return 0;
    return 1;
}
```

Funkcja: Znajdz indeks najmniejszego elementu

```
int indeks_min(int n, const int tab[]) {
    int min = tab[0], idx = 0;
    for (int i = 1; i < n; i++)
        if (tab[i] < min) { min = tab[i]; idx = i; }
    return idx;
}
```

Funkcja: Znajdz indeks najwiekszego elementu

```
int indeks_max(int n, const int tab[]) {
    int max = tab[0], idx = 0;
    for (int i = 1; i < n; i++)
        if (tab[i] > max) { max = tab[i]; idx = i; }
    return idx;
}
```

Funkcja: Oblicz sume indeksow liczb parzystych

```
int suma_ind_parz(int n, const int tab[]) {
    int s = 0;
    for (int i = 0; i < n; i++)
        if (tab[i] % 2 == 0) s += i;
    return s;
}
```

Funkcja: Oblicz iloczyn wszystkich dodatnich

```
int iloczyn_dodatnich(int n, const int tab[]) {
    int p = 1, ok = 0;
    for (int i = 0; i < n; i++)
        if (tab[i] > 0) { p *= tab[i]; ok = 1; }
    return ok ? p : 0;
}
```

Funkcja: Policz liczby wystepujace wiecej niz raz

```
int powtorzenia(int n, int tab[]) {
    int c = 0;
    for (int i = 0; i < n; i++)
        for (int j = i + 1; j < n; j++)
            if (tab[i] == tab[j]) { c++; break; }
    return c;
}
```

Funkcja: Ustaw 1 na miejscach o indeksach parzystych

```
void jedynki_parz_indeksy(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        if (i % 2 == 0) tab[i] = 1;
}
```

Funkcja: Ustaw -1 na miejscach o indeksach nieparzystych

```
void minusy_nieparz_indeksy(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        if (i % 2 != 0) tab[i] = -1;
}
```

Funkcja: Sprawdz czy tablica zawiera 0

```
int zawiera_zero(int n, int tab[]) {
    for (int i = 0; i < n; i++)
        if (tab[i] == 0) return 1;
    return 0;
}
```

Funkcja: Oblicz roznice miedzy max a min w tablicy

```
int roznica_max_min(int n, int tab[]) {
    int max = tab[0], min = tab[0];
    for (int i = 1; i < n; i++) {
        if (tab[i] > max) max = tab[i];
        if (tab[i] < min) min = tab[i];
    }
    return max - min;
}
```

Funkcja: Sprawdz czy tablica jest niemalejaca

```
int niemalejaca(int n, int tab[]) {
    for (int i = 1; i < n; i++)
        if (tab[i] < tab[i - 1]) return 0;
    return 1;
}
```