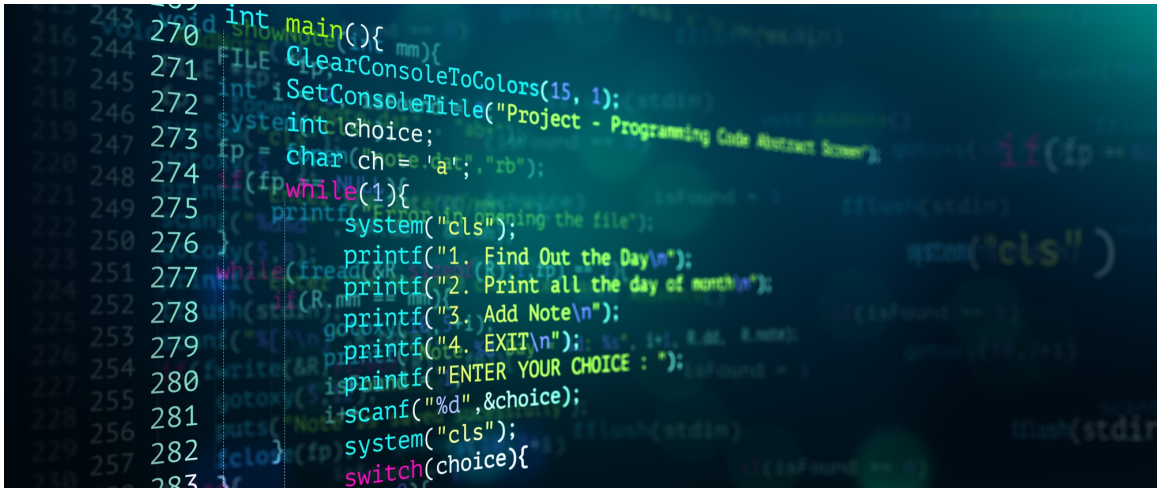

Zbiór zadań - C

Piotr Jastrzębski

2024-03-17

Spis treści

Zbiór zadań - C 3

I Podstawy języka C 4

1 Operacje wejścia, wyjścia. 5

2 Instrukcje warunkowe, operator warunkowy 6

3 Pętle 8

4 Funkcje 9

5 Wskaźniki 11

6 Wskaźniki na funkcję 14

7 Debugowanie 15

II Dodatki 17

8 Formatowanie zmiennych liczbowych w `printf` 18

9 Operatory arytmetyczne 20

10 Operatory bitowe 21

Bibliografia i inne zbiory zadań 22

Historia zmian 23

Zbiór zadań - C

Tu będzie zbiór zadań z programowania w języku C. Inspiracją było zebranie zadań powstałych w trakcie prowadzenia zajęć dydaktycznych realizowanych na Wydziale Matematyki i Informatyki Uniwersytetu Warmińsko-Mazurskiego w Olsztynie.

Rozwiązania wybranych zadań dostępne są tutaj: https://github.com/pjastr/zbior-zadan-c-rozw.

Cześć I

Podstawy języka C

1 Operacje wejścia, wyjścia.

1. Napisz program, który prosi użytkownika o wprowadzenie jednej liczby całkowitej, a następnie wyświetla ją na ekranie.
2. Stwórz program, który wczytuje od użytkownika dwie liczby zmiennoprzecinkowe i wypisuje ich różnicę.
3. Zaprojektuj aplikację, która pyta użytkownika o jego rok urodzenia, a następnie wypisuje rok poprzedni (o jeden mniejszy).
4. Napisz program, który wczytuje od użytkownika trzy liczby całkowite i wypisuje ich średnią jako wartość zmiennoprzecinkową.
5. Utwórz program, który prosi użytkownika o wprowadzenie dwóch liter (znaków), a następnie wypisuje je w odwrotnej kolejności.
6. Napisz program, który prosi użytkownika o wprowadzenie jednej liczby zmiennoprzecinkowej, a następnie podwaja jej wartość i wypisuje wynik.
7. Utwórz program, który wczytuje od użytkownika liczbę zmiennoprzecinkową reprezentującą kwotę w dolarach, a następnie wypisuje, ile to jest euro, przyjmując stały kurs wymiany (np. 1 dolar = 0.85 euro).
8. Napisz program, który wyświetla na ekranie tekst: `To jest cytat: "Często używam języka C."`. Upewnij się, że znaki cudzysłowu są poprawnie wyświetlane jako część napisu.
9. Stwórz program, który wypisuje na standardowe wyjście ścieżkę do folderu w systemie Windows, np. `C:\Program Files\MojaAplikacja\` (na sztywno, bez pobierania czegoś z systemu) lub `C:\\Program Files\\MojaAplikacja\\`.
10. Zaprojektuj program, który pokazuje, jak wypisać na ekranie następujący tekst: `Specjalne znaki: \t (tabulacja), \n (nowa linia), % (procent), \\ (ukośnik wsteczny).`.
11. Napisz program, który wczytuje ze standardowego wejścia dwie liczby wymierne reprezentujące długości boków trójkąta prostokątnego. Następnie oblicz i wyświetl długość przeciwprostokątnej.
12. Napisz program, który wczytuje ze standardowego wejścia całkowitą i wypisuje na standardowym wyjściu jej wartość bezwzględną.
13. Napisz program, który wczytuje ze standardowego wejścia zmiennoprzecinkową i wypisuje na standardowym wyjściu jej wartość bezwzględną.
14. Znajdź przykład i wyświetl na standardowym wyjściu, kiedy dodawanie liczb zmiennoprzecinowych nie jest łączne.

2 Instrukcje warunkowe, operator warunkowy

W poniższych zadaniach jako instrukcję warunkową należy wykorzystać `if`, `if else`, `switch case`. Przez operator warunkowy rozumie się `?`.

1. Napisz program, który prosi użytkownika o wprowadzenie liczby całkowitej. Program powinien wyświetlić informację, czy wprowadzona liczba jest dodatnia, ujemna czy równa zero.
2. Napisz program, który przyjmuje od użytkownika dwie liczby całkowite i wyświetla większą z nich.
3. Napisz program, który prosi o wprowadzenie oceny w skali od 1 do 5. Program powinien wyświetlić opis oceny: niedostateczny (1), dopuszczający (2), dostateczny (3), dobry (4), bardzo dobry (5). Dla liczby spoza zakresu, program powinien wyświetlić komunikat o błędzie.
4. Napisz program, który prosi użytkownika o wprowadzenie trzech różnych liczb całkowitych i wyświetla najmniejszą z nich.
5. Napisz program, który pyta użytkownika o rok i sprawdza, czy podany rok jest rokiem przestępnym. Rok przestępny to taki, który jest podzielny przez 4, ale nie jest podzielny przez 100, chyba że jest też podzielny przez 400.
6. Napisz program, który przyjmuje od użytkownika dwie liczby całkowite i wyświetla informację, czy suma obu liczb jest parzysta czy nieparzysta.
7. Napisz program, który przyjmuje od użytkownika trzy liczby zmiennoprzecinkowe `a, b, c`. Potrakuj je jako współczynniki równania kwadratowego $ax^2 + bx + c = 0$. Na standardowym wyjściu wypisz wszystkie warianty rozwiązań tego równania.
8. Napisz program, który przyjmuje dwie liczby całkowite jako wejście od użytkownika i używa operatora warunkowego, aby znaleźć i wyświetlić największą z nich.
9. Napisz program, który przyjmuje trzy liczby całkowite jako wejście od użytkownika i używa operatora warunkowego, aby znaleźć i wyświetlić najmniejszą z nich.
10. Używając operatora warunkowego `?`, napisz program, który przyjmuje od użytkownika jedną liczbę całkowitą i wyświetla "parzysta" lub "nieparzysta" w zależności od wartości liczby.

11. Stwórz program, który prosi o wprowadzenie dwóch liczb zmiennoprzecinkowych i używa operatora warunkowego, aby wyświetlić, która z nich jest większa, lub czy są równe z dokładnością do dwóch miejsc po przecinku.

12. Używając operatora warunkowego, napisz program, który prosi użytkownika o wprowadzenie oceny w skali od 0 do 100 i wyświetla "Zdane", jeśli ocena jest większa lub równa 51, lub "Nie zdane" w przeciwnym przypadku.

13. Napisz program, który przyjmuje rok jako wejście od użytkownika i za pomocą operatora warunkowego sprawdza, czy jest to rok przestępny. Program powinien wyświetlać "Rok przestępny" lub "Rok nieprzestępny" w zależności od wyniku.

3 Pętle

Zadania należy rozwiązać bez tablic, napisów, wskaźników, napisów, wbudowanych funkcji matematycznych. Nie twórz samodzielnie też własnych funkcji.

1. Napisz program, który wyświetla wszystkie liczby całkowite od 1 do 100, używając pętli `for`.
2. Utwórz program, który prosi użytkownika o wprowadzenie liczby całkowitej `n`, a następnie wyświetla sumę wszystkich liczb całkowitych od 1 do `n` używając pętli `while`.
3. Napisz program, który czyta od użytkownika liczby całkowite do momentu wprowadzenia zera i następnie wyświetla sumę wszystkich wprowadzonych liczb pozytywnych oraz sumę wszystkich liczb negatywnych, używając pętli `do-while`.
4. Stwórz program, który oblicza i wyświetla silnię podanej przez użytkownika nieujemnej liczby całkowitej, używając pętli `for`.
5. Napisz program, który wyświetla pierwszych 10 liczb ciągu Fibonacciego, używając pętli `while`.
6. Napisz program, który prosi użytkownika o wprowadzenie dodatniej liczby całkowitej `n`, a następnie oblicza i wyświetla $\lfloor \sqrt{n} \rfloor$ (część całkowita/podłoga pierwiastka kwadratowego).
7. Napisz program, który prosi użytkownika o wprowadzenie dodatniej liczby całkowitej `n`, a następnie oblicza i wyświetla $\lceil \sqrt{n} \rceil$ (sufit pierwiastka kwadratowego).
8. Napisz program, który prosi użytkownika o wprowadzenie 10 dodatnich liczb całkowitych i ustawia je w ciąg $a_1, \ldots, a_{10}$. Oblicz i wyświetl ile elementów ciągu spełnia nierówność $a_k < \frac{a_{k-1}+a_{k+1}}{2}$ dla $1 < k < 10$.
9. Napisz program, który prosi użytkownika o wprowadzenie dodatniej liczby całkowitej n, a następnie n dodatnich liczb całkowitych i ustawia je w ciąg $a_1, \ldots, a_n$. Oblicz i wyświetl ile elementów ciągu spełnia nierówność $a_k < \frac{a_{k-1}+a_{k+1}}{2}$ dla $1 < k < n$.

4 Funkcje

Zadania należy rozwiązać bez tablic, napisów, wskaźników, wbudowanych funkcji matematycznych. W zadaniach można stworzyć kilka funkcji pomocniczych. Do każdej funkcji z polecenia należy wywołać przypadek testowy (wywołać co najmniej jeden raz na poprawnym argumencie). W poleceniach używana jest konwencja `camelCase`. Zwróć uwagę, czy funkcja ma coś zwrócić czy wyświetlić.

1. Napisz funkcję `sumTwoNumbers`, której argumentami są dwie liczby całkowite. Funkcja ma wyświetlać sumę liczb przekazany jako argument funkcji. Stwórz przypadek testowy.
2. Napisz funkcję `calculateAbsoluteValue`, której argumentem jest liczba zmiennoprzecinkowa. Funkcja ma zwracać wartość bezwzględną liczby przekazanej jako argument funkcji. Stwórz przypadek testowy.
3. Napisz funkcję `calculateFactorial`, której argumentem jest liczba całkowita nieujemna. Funkcja ma zwracać wartość silni liczby przekazanej jako argument funkcji, obliczoną metodą nierekurencyjną. Stwórz przypadek testowy.
4. Napisz funkcję `sumNumbers`, której argumentem jest dodatnia liczba całkowita n. Funkcja ma zwracać sumę liczb od 1 do n włącznie. Stwórz przypadek testowy.
5. Napisz funkcję `sumSquares`, której argumentem jest dodatnia liczba całkowita n. Funkcja ma zwracać sumę kwadratów liczb od 1 do n włącznie. Stwórz przypadek testowy.
6. Napisz funkcję `calculatePowerOfTwo`, której argumentem jest liczba całkowita n. Funkcja ma zwracać wartość 2^n. Stwórz przypadek testowy.
7. Napisz funkcję `calculateSquareRootFloor`, której argumentem jest nieujemna liczba całkowita n. Funkcja ma zwracać część całkowitą pierwiastka kwadratowego z n. Stwórz przypadek testowy.
8. Napisz funkcję `countFunctionCalls`, która nie przyjmuje żadnych argumentów. Funkcja ma zliczać i wypisywać na standardowym wyjściu liczbę swoich wywołań od momentu uruchomienia programu. Stwórz przypadek testowy. Wykorzystaj zmienne statyczne.
9. Napisz funkcję `calculateFactorialRecursively`, której argumentem jest liczba całkowita nieujemna n. Funkcja ma zwracać wartość silni liczby n, obliczoną metodą rekurencyjną. Stwórz przypadek testowy.

10. Napisz funkcję `calculateFibonacciRecursively`, której argumentem jest liczba całkowita nieujemna n. Funkcja ma zwracać n-ty wyraz ciągu Fibonacciego, obliczony metodą rekurencyjną. Stwórz przypadek testowy.

11. Napisz funkcję `calculateArithmeticSequenceRecursively`, której argumentami są dwie liczby całkowite: dodatnie n (numer wyrazu ciągu do obliczenia) oraz d (różnica ciągu arytmetycznego), przy założeniu, że wyraz początkowy ciągu a_1 wynosi 1. Funkcja ma zwracać n-ty wyraz ciągu arytmetycznego, obliczony metodą rekurencyjną. Stwórz przypadek testowy.

12. Napisz funkcję `calculateGeometricSequenceRecursively`, której argumentami są dwie liczby całkowite: dodatnie n (numer wyrazu ciągu do obliczenia) oraz d (iloraz ciągu geometrycznego), przy założeniu, że wyraz początkowy ciągu a_1 wynosi 1. Funkcja ma zwracać n-ty wyraz ciągu geometrycznego, obliczony metodą rekurencyjną. Stwórz przypadek testowy.

13. Napisz funkcję `calculate13`, której argumentem jest dodatnia liczba całkowita n. Funkcja ma zwracać wartość wyrażoną wzorem $f(n) = 2f(n-1) + 3$, gdzie $f(1) = 1$. Stwórz przypadek testowy.

14. Napisz funkcję `calculate14`, której argumentem jest dodatnia liczba całkowita n. Funkcja ma zwracać wartość wyrażoną wzorem $f(n) = 3f(n-1) - 1$, gdzie $f(1) = 2$. Stwórz przypadek testowy.

15. Napisz funkcję `calculate15`, której argumentem jest dodatnia liczba całkowita n. Funkcja ma zwracać wartość wyrażoną wzorem $f(n) = f(n-1) + 2f(n-2)$, gdzie $f(1) = 1$ i $f(2) = 2$. Stwórz przypadek testowy.

16. Napisz funkcję `calculate16`, której argumentem jest dodania liczba całkowita n. Funkcja ma zwracać wartość wyrażoną wzorem $f(n) = 2f(n-1) + 3f(n-2)$, gdzie $f(1) = 2$ i $f(2) = 3$. Stwórz przypadek testowy.

17. Napisz rekurencyjną funkcję `calculate17`, której argumentem jest nieujemna liczba całkowita n. Funkcja ma zwracać wartość wyrażoną wzorem $f(n) = f(0) + f(1) + ... + f(n-1)$, gdzie $f(0) = f(1) = 1$. Stwórz przypadek testowy.

18. Napisz rekrurencyjną funkcję `calculateGCD`, której argumentami są dwie dodatnie liczby całkowite n i m. Funkcja ma zwracać największy wspólny dzielnik (NWD) tych liczb algorytmem Euklidesa. Stwórz przypadek testowy.

5 Wskaźniki

Zadania należy rozwiązać bez interpretacji wskaźników jako tablic.

1. Skopiuj lub przepisz kod i sprawdź wyniki na standardowym wyjściu:

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    printf("%Iu\n", sizeof(int));
    printf("%Iu\n", sizeof(int*));
    return 0;
}
```

2. W jednym pliku wykonaj czynności:

- Napisz funkcję `sum` z dwoma argumentami typu `int`. Funkcja ma zwracać sumę wartości przekazanych przez argumenty. Stwórz przypadek testowy.
- Napisz funkcję `sumVals` z dwoma argumentami, które są wskaźnikami do zmiennych typu `int`. Funkcja powinna zwracać sumę wartości, na które wskazują te argumenty. Utwórz przypadek testowy.
- Napisz funkcję `addPtr` z trzema argumentami, które są wskaźnikami do zmiennych typu `int`. Funkcja ma być procedurą (typ zwracany `void`). Funkcja ma ustawić wartość zmiennej wskazywanej przez trzeci argument funkcji jako sumę wartości wskazywanych przez dwa wcześniejsze argumenty. Utwórz przypadek testowy.

3. Napisz funkcję `copyInt` z argumentami: `x` typu `int` oraz `w`, który jest wskaźnikiem do `int`. Funkcja ma przepisać wartość `x` do zmiennej wskazywanej przez `w`. Stwórz przypadek testowy.
4. Napisz funkcję `findMax` z dwoma argumentami: wskaźnikiem `num1` na stałą wartość typu `double` i stałym wskaźnikiem `num2` na zmienną typu `double`. Funkcja ma zwracać większą wartość spośród tych, na które wskazują `num1` i `num2`. Utwórz przypadek testowy.

5. Napisz funkcję `initInts`, która nie przyjmuje argumentów i rezerwuje blok trzech zmiennych typu `int`, ustawiając ich wartości kolejno na 5, -12, 33. Funkcja zwraca wskaźnik na środkową zmienną. Utwórz przypadek testowy w funkcji `main`, który wyświetla wartości z bloku stworzonego przez funkcję.

6. Napisz funkcję `initFloats`, która nie przyjmuje argumentów i rezerwuje blok trzech zmiennych typu `float`, ustawiając ich wartości kolejno na 4.5, 2.3, -4.2. Funkcja zwraca wskaźnik na pierwszą ze zmiennych w bloku. Utwórz przypadek testowy w funkcji `main`, który wyświetla wartości z bloku stworzonego przez funkcję.

7. Napisz funkcję `initFlts`, która nie przyjmuje argumentów i rezerwuje blok czterech zmiennych typu `float`, ustawiając ich wartości kolejno na 0.5, 1.5, 2.5, i 3.5. Funkcja zwraca wskaźnik na ostatnią zmienną w bloku. Utwórz przypadek testowy w `main`, aby wyświetlić wartości z bloku stworzonego przez funkcję.

8. Napisz funkcję `sumToPtr` z trzema argumentami: dwoma wskaźnikami na stałe typu `int` i wskaźnikiem na zmienną typu `int`. Funkcja ma przepisać do zmiennej wskazywanej przez trzeci argument sumę wartości stałych wskazywanych przez pierwszy i drugi argument. Utwórz przypadek testowy.

9. Napisz funkcję `sqrCopy` z dwoma argumentami: wskaźnikiem na stałą typu `int` i wskaźnikiem na zmienną typu `int`. Funkcja ma przepisać kwadrat wartości stałej do zmiennej wskazywanej przez drugi argument. Utwórz przypadek testowy.

10. Napisz funkcję `subPtrs` z dwoma argumentami: wskaźnikiem `num1` na stałą wartość typu `double` i stałym wskaźnikiem `num2` na zmienną typu `double`. Funkcja ma zwracać różnicę wartości, na które wskazują `num1` i `num2`. Utwórz przypadek testowy.

11. Napisz funkcję `sumSqrs` z dwoma argumentami: wskaźnikiem `num1` na stałą wartość typu `int` i stałym wskaźnikiem `num2` na zmienną typu `int`. Funkcja ma zwracać sumę kwadratów wartości wskazywanych przez `num1` i `num2`. Utwórz przypadek testowy.

12. Napisz funkcję `linFuncVal` z trzema argumentami: wskaźnikiem `a` na stałą wartość typu `float`, stałym wskaźnikiem `b` na zmienną typu `float`, i wskaźnikiem `x` na stałą wartość typu `float`. Funkcja ma obliczać wartość funkcji liniowej `y=ax+b` dla argumentu `x`, gdzie `a` i `b` są wskazywane przez odpowiednie wskaźniki. Utwórz przypadek testowy.

13. Napisz funkcję `minPtr` z trzema argumentami, które są wskaźnikami na zmienne typu `int`. Funkcja zwraca wskaźnik na zmienną o najmniejszej wartości spośród tych, na które wskazują argumenty. Utwórz przypadek testowy w `main`, aby wyświetlić najmniejszą wartość spośród trzech zmiennych.

14. Napisz funkcję `multPtrs` z dwoma argumentami: wskaźnikiem `num1` na stałą wartość typu `double` i stałym wskaźnikiem `num2` na zmienną typu `double`. Funkcja zwraca iloczyn wartości wskazywanych przez te wskaźniki. Utwórz przypadek testowy.

15. Napisz funkcję `absVal` z jednym argumentem, którym jest wskaźnik na zmienną typu `int`. Funkcja oblicza wartość bezwzględną zmiennej wskazywanej przez wskaźnik i aktualizuje tę zmienną. Utwórz przypadek testowy w `main`, aby wyświetlić wartość zaktualizowanej zmiennej.

16. Napisz funkcję `swap` z dwoma argumentami: wskaźnikiem `ptr1` na zmienną typu `int` i wskaźnikiem `ptr2` na inną zmienną tego samego typu. Funkcja zamienia miejscami wartości wskazywane przez wskaźniki. Utwórz przypadek testowy.

17. Napisz funkcję `swapSign` z dwoma argumentami: wskaźnikiem `ptr1` na zmienną typu `double` i wskaźnikiem `ptr2` na inną zmienną tego samego typu. Funkcja zamienia miejscami wartości wskazywane przez wskaźniki, jeśli mają one różne znaki. W przeciwnym razie nie robi nic. Utwórz przypadek testowy.

6 Wskaźniki na funkcję

1. Napisz funkcję `calculate`, która przyjmuje dwa argumenty: wskaźnik na funkcję `operation` oraz liczbę całkowitą `number`. Funkcja `operation` ma przyjmować jeden argument typu `int` i zwracać wartość typu `int`. Funkcja `calculate` powinna wywołać funkcję `operation` z argumentem `number` i zwrócić jej wynik. Stwórz przypadek testowy.

2. Napisz funkcję o nazwie `applyFunction`, która przyjmuje trzy argumenty: wskaźnik na funkcję `func`, która przyjmuje jeden argument typu `int` i zwraca `int`, oraz dwie liczby całkowite: `start` i `end`. Funkcja `applyFunction` powinna wywołać funkcję `func` dla każdej liczby w zakresie od `start` do `end` (włącznie) i wydrukować wyniki na standardowe wyjście. Stwórz przypadek testowy.

3. Napisz funkcję, która otrzymuje trzy argumenty:

- dwa wskaźniki na funkcje z jednym argumentem typu `int` zwracające wartość typu `int`,
- dodatnią wartość `n` typu `int`,

i zwraca 1, jeżeli otrzymane w argumentach funkcje mają ten sam znak dla wartości dla liczb całkowitych od 0 do n, a zwraca 0 w przeciwnym wypadku. Stwórz przypadek testowy.

4. Napisz funkcję `calculateOperation`, która przyjmuje jako argumenty: wskaźnik na funkcję `operation`, która przyjmuje dwa argumenty typu `double` i zwraca `double`, oraz dwa argumenty typu `double - number1` i `number2`. Funkcja `calculateOperation` ma zwracać wynik wywołania funkcji `operation` na argumentach `number1` i `number2`. Stwórz przypadek testowy.

5. Napisz funkcję `modifyAndSum`, która ma przyjmować jako argument wskaźnik na funkcję `modifier`, która przyjmuje jeden argument typu `int` i zwraca `int`, oraz dwa argumenty typu `int`: `number1` i `number2`. Funkcja `modifyAndSum` powinna modyfikować obie liczby za pomocą funkcji `modifier` i zwracać ich sumę. Przykładem funkcji `modifier` może być funkcja, która zwiększa liczbę o 1 lub zmienia znak liczby. Stwórz przypadek testowy.

6. Stwórz funkcję `applyCondition`, która przyjmuje trzy argumenty: wskaźnik na funkcję `condition` zwracającą wartość typu `int` i przyjmującą `int`, wskaźnik na funkcję `action` również przyjmującą i zwracającą `int`, oraz wartość całkowitą `value`. Funkcja `applyCondition` powinna najpierw wywołać `condition` z `value` jako argumentem. Jeśli wynik to 1, `applyCondition` powinna następnie wywołać `action` na `value` i zwrócić wynik. W przeciwnym wypadku powinna zwrócić `value` bez zmian.

7 Debugowanie

1. Poniżej znajduje się kod w języku 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 złe wykonanie polecenia.

```c
#include <stdio.h>

int main() {
    int a = 10; // a = , b =
    int b = 5;  // a = , b =
    a = a + b;  // a = , b =
    b = a - b;  // a = , b =
    a = a - b;  // a = , b =
    b = a * b;  // a = , b =
    a = b / a;  // a = , b =
    b = a << 2; // a = , b =
    a = b >> 1; // a = , b =
    b = a & b;  // a = , b =
    a = a ^ b;  // a = , b =
    b = ~a;     // a = , b =
    return 0;
}
```

2. Poniżej znajduje się kod w języku 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 złe wykonanie polecenia.

```c
#include <stdio.h>

int main() {
    int a = 10; // a = , b = , c =
    int b = 5;  // a = , b = , c =
    int c = 0;  // a = , b = , c =
    a = a + b;  // a = , b = , c =
```

```
    b = a - b;  // a = , b = , c =
    a = a - b;  // a = , b = , c =
    c = a;      // a = , b = , c =
    a = b * c;  // a = , b = , c =
    b = a / c;  // a = , b = , c =
    c = b << 2; // a = , b = , c =
    b = c >> 1; // a = , b = , c =
    a = b & c;  // a = , b = , c =
    c = a ^ b;  // a = , b = , c =
    b = ~c;     // a = , b = , c =
    return 0;
}
```

Cześć II

Dodatki

8 Formatowanie zmiennych liczbowych w `printf`

1. Napisz program, który wyświetla liczbę całkowitą w formacie dziesiętnym.
2. Napisz program, który wyświetla liczbę całkowitą w formacie szesnastkowym z małymi literami.
3. Napisz program, który wyświetla liczbę całkowitą w formacie szesnastkowym z dużymi literami.
4. Napisz program, który wyświetla liczbę całkowitą w formacie ósemkowym.
5. Napisz program, który wyświetla liczbę całkowitą z wiodącymi zerami, zakładając, że szerokość pola wynosi 8 znaków.
6. Napisz program, który wyświetla liczbę całkowitą z wyrównaniem do prawej w polu o szerokości 10 znaków.
7. Napisz program, który wyświetla liczbę całkowitą z wyrównaniem do lewej w polu o szerokości 10 znaków.
8. Napisz program, który wyświetla dodatnią liczbę całkowitą z dodanym znakiem plus na początku.
9. Napisz program, który wyświetla liczbę całkowitą z użyciem notacji naukowej (z małymi literami).
10. Napisz program, który wyświetla liczbę całkowitą, dodając separator tysięcy.
11. Napisz program, który wyświetla liczbę zmiennoprzecinkową 123.456789 z dokładnością do dwóch miejsc po przecinku.
12. Stwórz program, który formatuje i wyświetla liczbę zmiennoprzecinkową -9876.54321, tak aby była przedstawiona z co najmniej 10 miejscami przed przecinkiem, dopełniając brakujące miejsca spacjami.
13. Utwórz program, który prezentuje liczbę zmiennoprzecinkową 0.000789 w notacji naukowej z dokładnością do czterech miejsc po przecinku.
14. Zaprojektuj program, który wyświetla liczbę zmiennoprzecinkową 12345.6789 z użyciem notacji naukowej i dokładnością do jednego miejsca po przecinku.
15. Stwórz program, który wyświetla liczbę zmiennoprzecinkową 3.14159 w formacie, gdzie przed liczbą znajduje się znak plus, z dokładnością do trzech miejsc po przecinku.
16. Napisz program, który formatuje i wyświetla liczbę zmiennoprzecinkową 123456.789 z dokładnością do pięciu miejsc po przecinku, zapewniając, że przed liczbą pojawi się miejsce na znak.
17. Utwórz program, który wyświetla liczbę zmiennoprzecinkową -0.0025 z dokładnością do sześciu miejsc po przecinku, zawsze zaczynając od znaku minus.

18. Zaprojektuj program, który przedstawia liczbę zmiennoprzecinkową 0.00123456789 w notacji naukowej z dokładnością do dwóch miejsc po przecinku i zawsze z przedrostkiem plus dla dodatnich wartości.
19. Stwórz program, który wyświetla liczbę zmiennoprzecinkową 9999999.99999 z dokładnością do trzech miejsc po przecinku, używając notacji naukowej, gdy jest to konieczne.
20. Napisz program, który formatuje i wyświetla liczbę zmiennoprzecinkową -123.456 w taki sposób, że zawsze zajmuje ona dokładnie 12 miejsc, w tym znak, liczby przed i po przecinku, dopełniając niewykorzystane miejsca spacjami.

9 Operatory arytmetyczne

1. Napisz program, który oblicza resztę z dzielenia sumy dwóch liczb całkowitych przez trzecią liczbę całkowitą.
2. Stwórz program, który oblicza różnicę kwadratów dwóch podanych liczb całkowitych.
3. Opracuj program, który oblicza iloczyn różnicy dwóch liczb całkowitych i trzeciej liczby całkowitej.
4. Zaprojektuj program, który oblicza średnią geometryczną bezwzględnych wartości trzech podanych liczb całkowitych.
5. Napisz program, który oblicza kwadrat sumy dwóch podanych liczb całkowitych.
6. Stwórz program, który oblicza sumę kwadratów trzech podanych liczb całkowitych.
7. Opracuj program, który oblicza, ile razy jedna podana liczba całkowita mieści się w drugiej podanej liczbie całkowitej.
8. Zaprojektuj program, który oblicza kwadrat różnicy dwóch podanych liczb całkowitych.
9. Napisz program, który oblicza iloraz sumy dwóch liczb całkowitych przez ich różnicę.
10. Stwórz program, który oblicza sumę trzech kolejnych liczb całkowitych, zaczynając od podanej liczby całkowitej.
11. Znajdź średnią arytmetyczną trzech liczb zmiennoprzecinkowych.
12. Oblicz wartość wyrażenia $\frac{1}{\frac{1}{a}+\frac{1}{b}+\frac{1}{c}}$, gdzie a, b, c są różne od zera.
13. Wylicz wartość e^x dla małych wartości x np. $\|x\| < 0.01$ przybliżając e^x jako $1 + x$, bez użycia funkcji eksponencjalnych.
14. Oblicz pole trójkąta o bokach a, b i kącie między nimi C w stopniach, używając wzoru $0.5 \cdot a \cdot b \cdot \sin C$, przyjmując $\sin C \approx C$ dla małych kątów w radianach.
15. Oblicz $\sqrt[3]{x}$ dla małych wartości x np. $\|x\| < 0.01$ przybliżając $\sqrt[3]{x} \approx 1 + \frac{1}{3}x$, bez użycia funkcji pierwiastkowania.

10 Operatory bitowe

1. Zamień wartości dwóch zmiennych całkowitych bez użycia dodatkowej zmiennej.
2. Sprawdź, czy liczba całkowita jest parzysta czy nieparzysta bez użycia instrukcji warunkowych.
3. Wyznacz wartość bitu na określonej pozycji w liczbie całkowitej.
4. Zeruj wartość bitu na określonej pozycji w liczbie całkowitej.
5. Odwróć wartość wszystkich bitów w liczbie całkowitej.
6. Przesuń bity liczby całkowitej o określoną liczbę pozycji w lewo.
7. Przesuń bity liczby całkowitej o określoną liczbę pozycji w prawo.
8. Wyznacz "XOR" dwóch liczb całkowitych.

Bibliografia i inne zbiory zadań

Krzaczkowski, Jacek. 2011. *Zadania z programowania w języku C/C++, cz. I.* Instytut Informatyki UMCS Lublin.

Historia zmian

- 17.02.2024 - dodano część: Formatowanie zmiennych liczbowych w `printf`, Operatory arytmetyczne, Operatory bitowe
- 25.02.2024 - dodano część: Instrukcje warunkowe, operator warunkowy, Pętle
- 3.03.2024 - dodano część: Funkcje
- 10.03.2024 - dodano część: Wskaźniki
- 11.03.2024 - dodano link do repozytorium na wybrane rozwiązania
- 17.03.2024 - dodano część: Wskaźniki na funkcję, Debugowanie