東京ITスクール
TOKYO IT SCHOOL

C 言語 練習問題 上級編

練習問題を行う環境

Visual Studio Community2022 を利用することを想定しています。

練習問題の解答例は、以下の環境でコンパイルし動作確認済みです。

環境:Visual Studio 2022 Developer Command Prompt v16.4.5
ソースコードの文字コード:UTF-8
コンパイル時のコマンドとオプション: cl /source-charset:utf-8

開発環境の設定により、コンパイルエラーやワーニングが発生する可能性もあることをご了承ください。

練習問題の注意事項

C 言語およびプログラミングの理解を深めること、アルゴリズムを考える力を養うことを目的として作成しています。

プログラムを記述する前に、処理の流れを考えることを心掛けてください。

目次

1. 条件分岐、反復、配列の活用

Question01_01

約数、素数であるか、完全数であるかを判定するプログラムを作成しましょう。

約数:正の整数(自然数)を割り切ることのできる自然数
素数:1 と自分自身でしか割り切れない自然数のこと
　1~30 のうち素数は、2,3,5,7,11,13,17,19,23,29
完全数: ある自然数 N の約数のうち、N を除く約数の総和が、N と一致する数
　1~30 のうち完全数は下記の 2 つ
　6: 約数:1,2,3,6　6 を除く約数の和 1+2+3=6
　28: 約数:1,2,4,7,14,28 28 を除く約数の和 1+2+4+7+14=28

下記、<プログラムの条件>に合うプログラムを作成し、実行結果を表示するコードを記述して、コンパイル、実行しましょう。

< プログラムの条件 >
・　フォルダ名、ファイル名は以下の指示に従うこと
　　ソリューション名:c_advanced_q01
　　プロジェクト名:c_advanced01_01
　　ファイル名:Qadv01_01.c

・正の整数の入力を受け付けること
・条件分岐、反復を利用すること
・必要に応じて変数を宣言し利用すること
・次ページの実行結果を表示するコードを記述すること

実行結果(入力値が完全数の場合)　太字部分:入力値

```
正の整数を入力してください ： 28

「28」の約数は 1 2 4 7 14 28
28 は素数ではありません！
28 は完全数です！
```

実行結果(入力値が完全数ではなく素数の場合)　太字部分:入力値

```
正の整数を入力してください ： 29

「29」の約数は 1 29
29 は素数です！
29 は完全数ではありません！
```

実行結果(入力値が完全数、素数ではない場合)　太字部分:入力値

```
正の整数を入力してください ： 30

「30」の約数は 1 2 3 5 6 10 15 30
30 は素数ではありません！
30 は完全数ではありません！
```

Question01_02

ツェラーの法則とうるう年の判定を利用して、入力された年月日の曜日を表示するプログラムを作成しましょう。

<ツェラーの法則 曜日を求める公式>
公式:(年+(年/4)-(年/100)+(年/400)+ (13×月+8)/5+日)を 7 で割った余り
公式で求められる 0 から 6 の数値を日曜日から順に当てはめることで曜日が算出できる。なお、1 月と 2 月は、前年の 13 月と 14 月として扱う

<うるう年の判定>
400 で割り切れるもしくは、4 で割り切れかつ 100 で割り切れない年

<プログラムの条件>
- ソリューション名、プロジェクト名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q01
  プロジェクト名:c_advanced01_02
  ファイル名:Qadv01_02.c

- 曜日の文字列を格納する配列を利用すること
- 日付を扱う標準ライブラリを利用しないこと
- 計算式、条件分岐、繰り返し文を利用すること
- 入力された月、日の値が実際に存在する月、日であることを判定すること
  - 月:1~12
  - 日:
    1,3,5,7,8,10,12 月の場合 1~31
    4,6,9,11 月の場合 1~30
    うるう年の 2 月の場合 1~29
    うるう年以外の 2 月の場合 1~28

- 下記の実行結果を表示するコードを記述すること
  実行結果 太字部分:入力値

```
入力された年月日の曜日を表示します。
年? 2021
月? 08
日? 31
2021 年 8 月 31 日は、火曜日です。
```

次ページに続く

実行結果(無効な月、日を入力した場合) 太字部分:入力値

```
入力された年月日の曜日を表示します。
年? 2021
月? 13
無効な値です。もう一度入力してください。
月? 12
日? 34
無効な値です。もう一度入力してください。
日? 32
無効な値です。もう一度入力してください。
日? 31
2021 年 12 月 31 日は、金曜日です。
```

Question01_03

Question01_02 で利用したツェラーの法則とうるう年の判定を利用して、入力された年月のカレンダーを表示するプログラムを作成しましょう。

<プログラムの条件>

- ソリューション名、プロジェクト名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q01
  プロジェクト名:c_advanced01_03
  ファイル名:Qadv01_03.c

- 日付を扱う標準ライブラリを利用しないこと
- 計算式、条件分岐、繰り返し文、配列を利用すること
- 入力された月の値が実際に存在する月(1~12)であることを判定すること
- 1 日(ついたち)の曜日を判定し、曜日に合わせて表示位置を調整すること

- 下記の実行結果を表示するコードを記述すること
  実行結果(9 月) 太字部分:入力値

```
入力された年月のカレンダーを表示します。
年? 2021
月? 9
-----2021 年 9 月-----
  日  月  火  水  木  金  土
                1   2   3   4
  5   6   7   8   9  10  11
 12  13  14  15  16  17  18
 19  20  21  22  23  24  25
 26  27  28  29  30
```

  実行結果(うるう年の 2 月の場合) 太字部分:入力値

```
入力された年月のカレンダーを表示します。
年? 2020
月? 2
-----2020 年 2 月-----
  日  月  火  水  木  金  土
                        1
  2   3   4   5   6   7   8
  9  10  11  12  13  14  15
 16  17  18  19  20  21  22
 23  24  25  26  27  28  29
```

Question01_04

Question01_03 で作成したカレンダーを表示するプログラムを改良して、前月末の日付、翌月の初めの日付を含めたカレンダーを表示するプログラムを作成しましょう。

<プログラムの条件>

- ソリューション名、プロジェクト名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q01
  プロジェクト名:c_advanced01_04
  ファイル名:Qadv01_04.c

- 日付を扱う標準ライブラリを利用しないこと
- 計算式、条件分岐、繰り返し文、配列を利用すること
- 入力された月の値が実際に存在する月(1~12)であることを判定すること
- 1 日(ついたち)の曜日が月曜日以降の場合は、前月末の日付を表示すること
- 月末最終日の曜日が日曜日～金曜日の場合は、翌月初めの日付を表示すること
- うるう年を考慮したカレンダーを表示すること

- 下記の実行結果を表示するコードを記述すること
  実行結果(1 日水曜日、月末最終日木曜日の場合) 太字部分:入力値

```
入力された年月のカレンダーを表示します。
年? 2021
月? 9
-----2021 年 9 月-----
  日  月  火  水  木  金  土
 29  30  31   1   2   3   4
  5   6   7   8   9  10  11
 12  13  14  15  16  17  18
 19  20  21  22  23  24  25
 26  27  28  29  30   1   2
```

  実行結果(うるう年の 3 月の場合) 太字部分:入力値

```
入力された年月のカレンダーを表示します。
年? 2024
月? 3
-----2024 年 3 月-----
  日  月  火  水  木  金  土
 25  26  27  28  29   1   2
  3   4   5   6   7   8   9
 10  11  12  13  14  15  16
 17  18  19  20  21  22  23
 24  25  26  27  28  29  30
 31   1   2   3   4   5   6
```

Question01_05

以下の条件の「数学パズル:小町算」のすべての解(12 通り)を求めるプログラムを作成しましょう。

<数学パズル:小町算の条件>
- 1~9 の数字を順に並べ、順番は変えない
- 並べた数字の間に+、-を補って計算結果が 100 になる式を抽出
- 数字の間には、何も入れない場合もある
- 先頭(1 の前)に-を補う式も含める

<プログラムの条件>
- ソリューション名、プロジェクト名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q01
  プロジェクト名:c_advanced01_05
  ファイル名:Qadv01_05.c

- 計算式、条件分岐、繰り返し文、配列を利用すること
- 下記のような実行結果になるように表示すること
  ただし、式の表示順序は、下記と異なってもよい

```
01:  - 1 + 2 - 3 + 4 + 5 + 6 + 78 + 9 = 100
02: 12 - 3 - 4 + 5 - 6 + 7 + 89 = 100
03: 123 - 4 - 5 - 6 - 7 + 8 - 9 = 100
04: 123 - 45 - 67 + 89 = 100
05: 123 + 4 - 5 + 67 - 89 = 100
06: 123 + 45 - 67 + 8 - 9 = 100
07: 12 + 3 - 4 + 5 + 67 + 8 + 9 = 100
08: 12 + 3 + 4 + 5 - 6 - 7 + 89 = 100
09: 1 + 23 - 4 + 56 + 7 + 8 + 9 = 100
10: 1 + 23 - 4 + 5 + 6 + 78 - 9 = 100
11: 1 + 2 + 3 - 4 + 5 + 6 + 78 + 9 = 100
12: 1 + 2 + 34 - 5 + 67 - 8 + 9 = 100
```

2. 構造体、ポインタ、関数の活用

Question02_01

社員の名前、役職、部署、在籍歴を表示するプログラムを作成してください。
下記<プログラムの条件>に合うプログラムを作成し、実行結果を表示するコードを記述して、コンパイル、実行しましょう。

< プログラムの条件 >

・ フォルダ名、ファイル名は以下の指示に従うこと
　ソリューション名:c_advanced_q02
　プロジェクト名:c_advanced02_01
　ファイル名:Qadv02_01.c

・ 構造体、ポインタ、関数を利用すること
・ 必要に応じて、配列や繰り返しも利用すること
・ 構造体は以下のものを作成すること
　① 部署の構造体

```
typedef struct
{
    char deptName[50];     // 部署名
    int  los;    // 勤続年数
}DEPT;
```

　② 社員の構造体

```
typedef struct
{
    char empName[256];     // 社員名
    char director[50];   //役職
    DEPT dept;          //部署の構造体

 }EMPLOYEE;
```

・ 社員情報

名前	役職	部署	在籍歴
高橋	課長	営業部	10
橋本	部長	人事部	3
真鍋	係長	開発部	8

・　以下の実行結果を表示するコードを記述すること

実行結果

```
社員情報を表示します
名前：高橋　　役職：課長　　部署：営業部　　在籍歴：10 年
名前：橋本　　役職：部長　　部署：人事部　　在籍歴：3 年
名前：真鍋　　役職：係長　　部署：開発部　　在籍歴：8 年
```

<ヒント>

処理の流れが考えられない方は参考にしてみましょう。

① 構造体を作成する

② main()の処理を考える

(ア)EMPLOYEE(構造型)の配列を用意し、初期化

(イ)社員情報を表示します。という文字列を出力(printf())

(ウ)引数に(ア)の配列を入れた状態で、③の関数を呼び出す

③ 社員情報を表示する関数を用意する

② の(ア)の配列に入っている社員情報を繰り返し処理を使って、表示する

Question02_02

3 人の学生の名前、3 教科(国数英)の合計点、成績を表示するプログラムを作成しましょう。下記<プログラムの条件>に合うプログラムを作成し、実行結果を表示するコードを記述して、コンパイル、実行しましょう。

< プログラムの条件 >

- フォルダ名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q02
  プロジェクト名:c_advanced02_02
  ファイル名:Qadv02_02.c
- 構造体、ポインタ、関数を利用すること
- 必要に応じて変数や関数を宣言し利用すること
- 学生の名前、各教科の点数(100 点満点)は、自由に入力できる仕様であること
- 成績は、3 教科の合計点を下記の評価に基づき決定すること
  合計点：① 240 点以上=A
  ② 150 点以上=B
  ③ 90 点以上=C
  ④ 89 点以下=D
- 以下の実行結果を表示するコードを記述すること

- 実行結果 太字部分:入力値

```
3 人分のデータを入力します。
国語、数学、英語の順で点数を入力してください。
名前>田中
1 教科目>100
2 教科目>100
3 教科目>100
1 人目の入力が完了しました。

名前>林
1 教科目>56
2 教科目>82
3 教科目>20
2 人目の入力が完了しました。

名前>佐々木
1 教科目>30
2 教科目>38
3 教科目>20
3 人目の入力が完了しました。

成績を表示します。
名前|3 教科合計|成績
田中      300       A
林        158       B
佐々木    88        D
```

Question02_03

構造体とポインタを使って、スタックを扱うプログラムを作成しましょう。

スタック:
基本的なデータ構造の一つで、要素が入ってきた順に一列に並べ、後に入れた要素から順に取り出すという規則で出し入れを行うもの。

下記、<プログラムの条件>に合うプログラムを作成し、実行結果を表示するコードを記述して、コンパイル、実行しましょう。

< プログラムの条件 >

- フォルダ名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q02
  プロジェクト名:c_advanced02_03
  ファイル名:Qadv02_03.c

- 正の整数の入力を受け付けること
- 構造体、ポインタ、関数を利用すること
- 必要に応じて変数や関数を宣言し利用すること
- 入力された数値をスタックに格納、0 が入力されたら、スタックから順に取り出し、表示すること
- 下記の役割を持つ 2 つの関数を必ず用意すること
  - pushStack スタックへの値の格納
  - popStack スタックからの値の取り出し

・次ページの実行結果を表示するコードを記述すること

実行結果 太字部分:入力値

```
99 までの整数値を入力してください。(0 を入力すると終了します。)
>34
---現在のスタックの状態---
#  34
------
1～99 までの整数値を入力してください。(0 を入力すると終了します。)
>45
---現在のスタックの状態---
#  45
#  34
------
1～99 までの整数値を入力してください。(0 を入力すると終了します。)
>67
---現在のスタックの状態---
#  67
#  45
#  34
------
1～99 までの整数値を入力してください。(0 を入力すると終了します。)
>86
---現在のスタックの状態---
#  86
#  67
#  45
#  34
------
1～99 までの整数値を入力してください。(0 を入力すると終了します。)
>0
入力された値をスタックから取り出して表示します。
POP で取り出した値:       86
---現在のスタックの状態---
#  67
#  45
#  34
------
POP で取り出した値:       67
---現在のスタックの状態---
#  45
#  34
------
POP で取り出した値:       45
---現在のスタックの状態---
#  34
------
POP で取り出した値:       34
---現在のスタックの状態---
空っぽです。
------
```

Question02_04

構造体とポインタを利用して双方向リストを扱うプログラムを作成しましょう。

双方向リスト:
基本的なデータ構造の一つで、データ要素が前後の要素へのリンク情報を保持しており、順方向、逆方向に要素を順に取り出すことができるもの。

下記、<プログラムの条件>に合うプログラムを作成し、実行結果を表示するコードを記述して、コンパイル、実行しましょう。

< プログラムの条件 >
・ フォルダ名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q02
  プロジェクト名:c_advanced02_04
  ファイル名:Qadv02_04.c

・ 正の整数の入力を受け付けること
・ 構造体、ポインタ、関数を利用すること
・ 必要に応じて変数や関数を宣言し利用すること
・ 入力された数値をリストに格納、0 が入力されたら、リストから入力された順と逆順に数値を取り出し、表示すること

・以下の実行結果を表示するコードを記述すること
実行結果 太字部分:入力値

```
整数値を入力してください。(0 を入力すると終了します。)
>12
整数値を入力してください。(0 を入力すると終了します。)
>34
整数値を入力してください。(0 を入力すると終了します。)
>56
整数値を入力してください。(0 を入力すると終了します。)
>78
整数値を入力してください。(0 を入力すると終了します。)
>0
次の順で数値が入力されました。
12      34      56      78
入力順と逆順に並べると
78      56      34      12
```

Question02_05

スタックと双方向リストを使って、中置記法を逆ポーランド記法(後置記法)に置き換えるプログラムを作成しましょう。

中置記法:演算子を計算対象の間に記術する方式 例:3+4×5
逆ポーランド記法:演算子を計算対象の後ろに記述する方式 例:3 4 5×+

< 数式の条件 >
・ 1~2 桁の数値と四則演算で構成される数式
・ 数式には()を含まない

下記、<プログラムの条件>に合うプログラムを作成し、実行結果を表示するコードを記述して、コンパイル、実行しましょう。

< プログラムの条件 >
・ フォルダ名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q02
  プロジェクト名:c_advanced02_05
  ファイル名:Qadv02_05.c
・ 構造体、ポインタ、関数を利用すること
・ 必要に応じて変数や関数を宣言し利用すること
・ 入力された数式を分析し中置記法から後置記法に変換し表示すること
・ 以下の実行結果を表示するコードを記述すること

実行結果 太字部分:入力値

```
() を含まない四則演算の計算式を入力してください。
(たし算:+、引き算:-、掛け算:*、割り算:/)
>12*3+45/9
入力された式:12*3+45/9
入力された式を逆ポーランド記法に置き換えました
12 3 * 45 9 / +
```

Question02_06

Question2_05 で作成した中置記法を逆ポーランド記法に置き換えるプログラムに計算処理を加えて、計算式を入力し、計算結果を表示するプログラムを作成しましょう。

中置記法:演算子を計算対象の間に記術する方式 例:3+4×5
逆ポーランド記法:演算子を計算対象の後ろに記述する方式 例:3 4 5×+

< 数式の条件 >
・ 1~2 桁の数値と四則演算で構成される数式
・ 数式には()を含まない

下記、<プログラムの条件>に合うプログラムを作成し、実行結果を表示するコードを記述して、コンパイル、実行しましょう。

< プログラムの条件 >
・ フォルダ名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q02
  プロジェクト名:c_advanced02_06
  ファイル名:Qadv02_06.c
・ 構造体、ポインタ、関数を利用すること
・ 必要に応じて変数や関数を宣言し利用すること
・ 入力された数式を分析し計算結果を導くこと
・ 計算結果は、小数点第 2 位まで表示すること
・ 以下の実行結果を表示するコードを記述すること

実行結果 太字部分:入力値

```
()を含まない四則演算の計算式を入力してください。
数値は 2 桁までの整数、演算子 たし算:+、引き算:-、掛け算:*、割り算:/
>12*3+45/6-7
12*3+45/6-7=36.50
```

Question02_07

テンパズルという 4 つの数字と四則演算を用いて 10 になる式を導くプログラムを作成しましょう。

<テンパズルの条件>
- 入力された 4 つの数字を使用する
- 4 つの数字と演算子を使って、一桁の数値で構成される計算式を生成する
- 演算子は、+,-,*, /の 4 つのみを利用する
- ()は利用しない
- 数字の順番を並べかえを認める
- 計算結果が 10 になるすべてのパターンを抽出する
- 割り算の結果は、小数点以下を切り捨てた値を扱うこと

下記、<プログラムの条件>に合うプログラムを作成し、実行結果を表示するコードを記述して、コンパイル、実行しましょう。

< プログラムの条件 >
- フォルダ名、ファイル名は以下の指示に従うこと
  ソリューション名:c_advanced_q02
  プロジェクト名:c_advanced02_07
  ファイル名:Qadv02_07.c
- 構造体、ポインタ、関数を利用すること
- 必要に応じて変数や関数を宣言し利用すること
- 入力された数値を使って 10 になる計算式をすべて表示すること
- 以下の実行結果を表示するコードを記述すること

実行結果 (該当する式がある場合) 太字部分:入力値

```
4 つの数字を入力してください。(入力例:1234)
>5679
式番号 :01        5 + 6 - 9 / 7 = 10
式番号 :02        5 * 6 / 9 + 7 = 10
式番号 :03        5 - 9 / 7 + 6 = 10
式番号 :04        6 + 5 - 9 / 7 = 10
式番号 :05        6 * 5 / 9 + 7 = 10
式番号 :06        6 - 9 / 7 + 5 = 10
式番号 :07        7 + 5 * 6 / 9 = 10
式番号 :08        7 + 6 * 5 / 9 = 10

■■■ 式のパターン数 : 8

終了
```

次のページへ続く

実行結果（該当する式がない場合）太字部分:入力値

```
4つの数字を入力してください。(入力例:1234)
>6789

■■■ 該当する式はありません

終了
```