

第10章

プログラミングに チャレンジ!



最後の章に練習問題を載せておきます。

簡単な問題そして少し難しい問題等ありますので、復習のつもりでチャレンジしてみてください。

これらの問題を解くことができれば、もうポイントは卒業です。

10-1 練習問題を解いてみよう



9つの練習問題

本書で学んだ内容を確認してみましょう。いくつか練習問題を載せておきますので、理解度を確認するためにチャレンジしてみてください。解答例は10-2節にまとめてありますので参照してください。

問題 1

次のプログラムを実行すると何が表示されるか考えてみてください。

▼Q01.c

```
#include <stdio.h>

int main() {
    printf("%d\n", sizeof(long long));

    return 0;
}
```

問題 2

次のプログラムを実行すると何が表示されるでしょうか。

▼Q02.c

```
#include <stdio.h>

int main() {
    char szData[] = "abcde";

    printf("%d\n", sizeof(szData));

    return 0;
}
```

問題 3

次のプログラムを実行すると何が表示されるでしょうか。

▼Q03.c

```
#include <stdio.h>

void sizePrint(char szData[]);

int main() {
    char szData[] = "abcde";

    sizePrint(szData);

    return 0;
}

void sizePrint(char szData[]) {
    printf("%d\n", sizeof(szData));

    return;
}
```

問題 4

次のプログラムを実行するとpiDataの値は0x0028FEE4と表示されました。このときpiData2の値はどう表示されるでしょう。

▼Q04.c

```
#include <stdio.h>

int main() {
    int i = 100;
    int *piData = NULL;
    int *piData2 = NULL;
```



```

piData = &i;

printf("piDataの値: 0x%p\n", piData);

piData2 = piData + 1;

printf("piData2の値: 0x%p\n", piData2);

return 0;
}

```

問題5

文字列の長さを返すmyStrlen関数を作成してください（この場合の文字列の長さは末尾のNULL文字を含まない長さとしします）。
関数のプロトタイプ宣言は次の形とします。

```
int myStrlen(char *szData);
```

下記プログラムの [1] のエリアのコードを作成してください。

▼Q05.c

```

#include <stdio.h>

int myStrlen(char *szData);

int main() {
    char szData[] = "abcde";

    printf("szData 文字列の長さは: %d\n", myStrlen(szData));

    return 0;
}

```

```

}

int myStrlen(char *szData) {
    [1]
}

```

問題6

文字列として定義されているアルファベットの文字を、すべて大文字に変換して表示するプログラムを作成してください。ただし、処理コードにはポインタの書き方を使ってください（下記プログラムの [1] のエリアのコードを作成してください）。
文字列の長さを調べる場合は、string.hをインクルードしstrlen関数を利用してください。

また、大文字変換ではctype.hをインクルードしtoupper関数を利用することとします。

toupper 関数の仕様

```
int toupper(int c);
```

受け取ったcの値がアルファベット小文字のコードであれば、それを大文字のコードに変換して返す。

▼Q06.c

```

#include <stdio.h>
#include <string.h>
#include <ctype.h>

int main() {
    char szData[] = "abcde";
    int i;

    printf("szData 文字列の変換前: %s\n", szData);

    [1]
}

```

```
printf("szData 文字列の変換後: %s\n", szData);

return 0;
}
```

問題7

現在日時をYYYY/MM/DD hh:mm:ssの形式で表示するプログラムを作成してください。

C言語には現在時間を取得するためのtime関数があります。この関数はtime.hをインクルードすることにより利用可能となります。

time関数で取得できるのは世界標準時間なので、これはlocaltime関数で日本時間に変換する必要があります。

変換された情報は、localtime関数が返すtm構造体の中に格納されます。

time関数の仕様

```
time_t time(time_t *ptime_tData);
```

現在の日時をtime_t型の値で返す。引数のポインタにも同じ値を返す。引数をNULLにした場合は、戻り値のみで値を返す。

localtimeの仕様

```
struct tm *localtime(const time_t *time_tData);
```

引数でtime_t型の日付データを渡すと、それを日本時間に変換してstruct tm構造体に格納し、そのポインタを返す

struct tm構造体の仕様

```
struct tm {
    int tm_sec; // 秒: 0-59
```

```
int tm_min; // 分: 0-59
int tm_hour; // 時間: 0-23
int tm_mday; // 日: 1-31
int tm_mon; // 月: 0-11 (0からなので1を足す必要あり)
int tm_year; // 年 1900年からの年なので1900を足す必要あり
int tm_wday; // 曜日 日曜日が0で順番の番号になっている (土曜日が6)
int tm_yday; // 1月1日からの日数: 0-365
int tm_isdst; // 夏時間フラグ (0: 夏時間でない、正: 夏時間、負: 不明)
};
```

上記情報を参考に、下記プログラムの [1] のエリアのコードを作成してください。

▼Q07.c

```
#include <stdio.h>
#include <time.h>

// プロトタイプ宣言
void printTime(struct tm *pstTm);

int main(void) {
    time_t time_tData;
    struct tm *pstTm;

    // 現在時刻を取得
    time_tData = time(NULL);

    pstTm = localtime(&time_tData); // 日本時間に変換

    printTime(pstTm);

    return 0;
}

void printTime(struct tm *pstTm) {
```


[1]

```

return;
}

```

問題8

引数で渡したファイル名からそのファイルのサイズを計算して表示するプログラムを作成してください。

```

C:¥CLang>Q08 Q08.c
ファイルのサイズ: 798 バイト

```

問題9

2つの文字列とそれを結合したデータを格納するメモリを準備し、それらを関数に渡して文字列を結合させる関数を作成してください。

下記のプログラムの [1] のエリアのコードを作成してください。

▼ Q09.c

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// プロトタイプ宣言
void addString(char *szData1, char *szData2, char *szAddArea);

int main(void) {
    char *szData1 = "abcd";
    char *szData2 = "efgh";
    char *szAddArea = NULL;

```

```

szAddArea = malloc(strlen(szData1) + strlen(szData2) + 1);
if (szAddArea == NULL) {
    printf("mallocエラー¥n");
    return -1;
}

addString(szData1, szData2, szAddArea);

printf("AddString:%s<¥n", szAddArea);

free(szAddArea);

return 0;
}

void addString(char *szData1, char *szData2, char *szAddArea) {

    [1]

    return;
}

```