小テスト

準備　プログラムを作る前に、以下の操作をしてファイルの準備をしておくこと。

1. GitHub から自分のリポジトリを clone しておく（既に今日の授業で実行済みの場合は不要）
   ```
   $  mygitclone  「自分の GitHub ユーザ名」
   $  cd prog3i-(ユーザ名)
   $  ./myconf
   ```
2. 今回の小テスト用のフォルダを作って移動する
   ```
   $ cd ~/prog3i-(ユーザ名)
   $ mkdir test106
   $ cd test106
   ```
3. テキストエディタでプログラムを開き、**先頭行に C のコメントとして自分の番号と名前を書く**
   ```
   $ gedit test106.c &
   ```

【問 1】　「引数で与えられた車のナンバーが正の値の場合はそのまま追加し、負の場合は正の値に変えて追加する」関数 add_car4() を作成してください。この関数のプロトタイプ宣言は以下のようになります。

```
void add_car4(Car *p, int n, double g);
  /* 課題 5-1 で作成した関数において、以下のような条件を追加する */
  /* まず、領域を確保して… */
  /* ・n が正の値の場合は、そのまま num に代入する  */
  /* ・n が負の値の場合は、n を正の値に変えて num に代入する  */
  /* g はそのままメンバ gas に代入する */
```

main() で動作を確認してください。動作の確認には、関数 show_carlist() も必要になります。

```
[main() での処理]
  Car head4;
  head4.num = 0; head4.gas = 0; head4.next = NULL;
  show_carlist(&head4, "head4 (1)");
  add_car4(&head4, 1357, 40.3);
  add_car4(&head4, -2468, 33.8);
  add_car4(&head4, 0, 26.1);
  show_carlist(&head4, "head4 (2)");
[実行例]
  --- head4 (1) ---
  num: 0, gas: 0.000000
  --- head4 (2) ---
  num: 0, gas: 0.000000
  num: 0, gas: 26.100000        (← 0 の場合はそのまま追加される)
  num: 2468, gas: 33.800000     (←負の値が、正の値に変換されている)
 num: 1357, gas: 40.300000
```

（20 点）

小テストの注意点

- 他人の力は借りずに、自分だけでプログラムを作成する。つまり、**通常の定期試験と同様**。

小テスト中に参照できるもの

- 教科書と配付資料
- 自分のホームディレクトリ（ホームフォルダ）以下に保存されているファイル

* **上記以外の情報を参照することは不正行為とする**
（例：USB で接続された機器に保存されているファイルの参照, ネットワークを介した情報の参照など）

答案の提出

1. **提出する全てのファイルの先頭行に、C のコメントとして自分の番号と名前を書く**
2. 端末内で、以下のコマンドで課題を提出

```
$ git add -A
$ git commit -m "小テスト 6 提出"
$ git push origin master
```

小テストの模範解答

```
/* 自分の番号と名前をここに書く */
#include <stdio.h>
#include <stdlib.h>

/* 車の情報を持つ構造体の宣言 */
typedef struct Car {
    int num;
    double gas;
    struct Car *next;
} Car;

/* 関数のプロトタイプ宣言 */
void show_carlist(Car *start, char *str);
void add_car4(Car *p, int n, double g);

/* リストの要素を出力する */
void show_carlist(Car *start, char *str)
{
    Car *pcar;
    printf("--- %s ---\n", str);
    for(pcar = start; pcar!=NULL;
        pcar = pcar->next) {
        printf("num: %d, gas: %lf\n",
               pcar->num, pcar->gas);
    }
}

/* リストに要素を追加する (その 4) */
void add_car4(Car *p, int n, double g)
{
    Car *new;
    /* 構造体 Car の領域を確保する */
    new = (Car *)malloc(sizeof(Car));
    /* 確保した領域のメンバに引数の値を代入する */
    if(n > 0) {
        new->num = n;
    } else {
        new->num = -1 * n;
    }
    new->gas = g;
    /* 確保した領域の next を更新する */
    new->next = p->next;
    /* p の next は確保した領域を参照する */
    p->next = new;
}

int main(void)
{
    Car head4;
    /* add_car4() の動作確認 */
    head4.num = 0; head4.gas = 0; head4.next = NULL;
    show_carlist(&head4, "head4 (1)");
    add_car4(&head4, 1357, 40.3);
    add_car4(&head4, -2468, 33.8);
    add_car4(&head4, 0, 26.1);
    show_carlist(&head4, "head4 (2)");

    return 0;
}
```