PART 01 OF 02

Ex1: define struct Triangle which has 3 sides (a, b, c) as members (float)

Implement function ***int isTriangle(Triangle);*** which accepts a variable of Triangle

The function returns 1 if 3 sides (a, b, c) can form a triangle, otherwise returns 0

Initialize a variable of Triangle in main() and call **isTriangle**() to get results

Ex2: using struct Triangle in ex1, implement function ***int isRightTriangle(Triangle);*** which accepts a variable of Triangle

The function returns 1 if 3 sides (a, b, c) can form a triangle and it is a right triangle which has one angle equal to 90 degrees. (You should use **isTriangle**() in ex1 to check if it is a triangle before checking if it is a right triangle)

Initialize a variable of Triangle in main() and call **isRightTriangle**() to get results

Ex3: define struct Student as following:

```
typedef struct {
    char name[20];
    int eng; //English grade [0, 100]
    int maths; //maths grade [0, 100]
} Student;
```

a. Initialize an array of 5 students with names and English/maths grades
   Print all students who have the maximum total of English and maths

b. English grade of each student is increased by 30.
   Then if English grade is greater than 100, English grade becomes 100.
   Print all students who has maximum English grade.

PART 02 OF 02

Ex4: define a struct named Car which has the following members

- Name, //car name
- Year, //manufacturing year
- Price, //selling price of a brand-new car
- MaxSpeed

You have to decide data type for each Car's member.

Ex5:

"In C programming, using scanf() and gets() in a loop to read multiple strings with spaces from keyboard sometimes produce unexpected results. In order to read multiple strings with spaces in a loop, you may need to write your own scanning function.

The following function is one way to properly read multiple strings with spaces in a loop.

```
void scanstr(char[]);
void scanstr(char str[]){
	char t;
	scanf("%c", &t); // to clear buffer
	scanf("%[^\n]s", str);
}
```

You can use the above scanstr() in the following exercises."

Implement function ***void scanYear(Car*);*** to ask users type in an integer for manufacturing year of a car.

The year must be from 1980 to 2019, so if users type in invalid year, the function should keep asking for correct input

In the end, the function should update the year of the car with valid input

Sample run:

```
Year: 1970
Year must be from 1980 to 2019
Year: 2020
Year must be from 1980 to 2019
Year: 1990
```

Ex6: Similar to Ex5, implement functions ***void scanPrice(Car*)*** with a validation such that Price must be from $30,000 to $400,000

In the end, the function should update the price of the car with valid input

Ex7: in main(), initialize a Car array of 02 elements and ask for user's inputs of all cars in the array. You must use **scanYear**() and **scanPrice**() functions above to validate and get inputs of year and price.

(advanced: implement function ***void getCarInputs(Car*, int);*** to ask for user's inputs of all cars in the array. Input Car* is the pointer to car array)

Sample run

```
Car[0]:
Name: Toyota
Year: 1970
Year must be from 1980 to 2019
Year: 2020
Year must be from 1980 to 2019
Year: 1990
Price: 20000
Price must be from $30,000 to $400,000
Price: 80000
MaxSpeed: 250
Car[1]:
….
….
-----------------------------------------
```

Ex8: implement function ***void printCarList(Car[], int);*** to print all cars in the array

Sample run

```
No#    |Name        |Year |Price       |MaxSpeed
1      |Toyota      |2000 |47000       |125
2      |Ford        |1995 |34000       |150
----------------------------------------
```

Ex9: implement function ***void printCarAfter2000(Car[], int);*** to print all cars in the array which were manufactured after 2000

Sample run should look similar to ex5

Ex10: define struct Stats {max, min, avg}

Implement function ***Stats getStatistics(Car[], int);*** to collect maximum price, minimum price and average price of all cars in the array and return all 3 prices in a Stats

S11p04