Übung 6 infT

1.
Gegeben ist das Header-File *dynComplex.h*. Schreiben Sie das Implementations-File *dynComplex.c*. Die Implementationen sollen so realisiert sein, dass nicht getestet wird, ob Argumente allenfalls **`NULL`** sind (statt auf **`Complex`**-Variablen zu zeigen); Application Programmer sind also selbst dafür verantwortlich, dass keine **`NULL`**-Pointer übergeben werden. Im File *dynComplexTest.c* sollen die Funktionen getestet werden.

```
/* dynComplex.h */
#ifndef DYNCOMPLEX_H
#define DYNCOMPLEX_H

typedef struct _Complex {
  double re;
  double im;
} Complex;

Complex *createComplex(double re, double im);
Complex *cloneComplex(Complex *z);
void deleteComplex(Complex *z);
Complex *addComplex(Complex *z1, Complex *z2);
Complex *subComplex(Complex *z1, Complex *z2);
Complex *negComplex(Complex *z);
Complex *multComplex(Complex *z1, Complex *z2);
Complex *conjComplex(Complex *z);
Complex *divComplex(Complex *z1, Complex *z2);
void printComplex(char *p, Complex *z);

#endif
```

Hinweise:
- Die Funktion **`createComplex`** erzeuge auf dem Heap eine komplexe Zahl, die mit den übergebenen Parametern initialisiert wird.
- **`cloneComplex`** lege eine Kopie auf dem Heap an.
- **`deleteComplex`** entferne die Zahl vom Heap.
- **`negComplex`** liefere von einer Ausgangszahl eine Zahl, deren Re- und Im-Teil je mit -1 multipliziert sind.
- **`printComplex`** diene für Aufrufe wie:
  `printComplex("zp1", zp1);`
  mit Ausgabe an **`cout`** der folgenden Art:
  `zp1 = 3.00 + 4.00 * i`

2.
Gegeben ist die Struktur **`Point`**, die ein Integer-Member **`x`** und ein Integer-Member **`y`** aufweise. Eine weitere Struktur **`CommentedPoint`** weise ein Member **`Point`** und ein Member Zeiger auf **`char`** auf.

a) Realisieren Sie die beiden Strukturen (unter Mitverwendung von **`typedef`**).

b) Schreiben Sie eine Funktion
`CommentedPoint *createCommentedPoint(int x, int y, char *commentp)`, die ein initialisiertes Objekt auf dem Heap anlegt. Der Aufruf der Funktion sehe z.B. so aus:
```
CommentedPoint *cp = createCommentedPoint(11, -3, "Schwerpunkt");
```
Merke:
Auch das via Argument angesprochene **`char`**-Array (hier also **`Schwerpunkt`**) soll auf den Heap kopiert werden.

c) Schreiben Sie nun auch die Funktionen **`cloneCommentedPoint`** und **`deleteCommentedPoint`**.

d) Schreiben Sie ein **`main`**, das in einem (dynamischen) Array mit Elementen Pointer auf **`CommentedPoint`** so viele **`CommentedPoint`**-Objekte anlegt, wie Sie von der Tastatur Zeilen eingeben. Der Abbruch der Eingabe erfolge mit einer Leerzeile.

e) Erweitern Sie **`main`** so, dass alle Objekte die Werte ihrer Member an **`cout`** ausgeben und dann die **`CommentedPoint`**s sowie das dynamische Array vom Heap entfernt werden.