1. Implementieren Sie ein Makro `B(` *bitfolge* `)`, das es ermöglicht, ganzzahlige Werte im Binärsystem zu kodieren. Das Makro soll in einer Headerdatei `binary.h` definiert werden.

   Bei Verwendung des GNU C-Compilers (Überprüfen Sie hierzu, dass ausschließlich das Makro `__GNUC__`, jedoch kein Makro `__clang__` existiert.) sollen Binärkonstanten, die mit dem Präfix `0b` beginnen und als Spracherweiterung bei diesem Compiler zur Verfügung stehen, die Aufrufe des Makros `B` ersetzen.

   Bei anderen Compilern sollen Aufrufe von `B` in geeignete Ausdrücke umgesetzt werden, die den durch *bitfolge* spezifierten Wert als Ergebniswert liefern. (Beispielsweise könnte auch eine selbstgeschriebene Funktion den gewünschten Ergebniswert liefern. Warum sollte in diesem Fall *bitfolge* als Zeichenkette an die Funktion übergeben werden?)

   Testen Sie Ihr Programm auch mit dem Clang Compiler (Aufruf: `clang`).

   Beispiel:

```
...
#include "binary.h"

int main( void ) {
  unsigned char a[10];

  a[0] = B(11111111);  /* 255     */
  a[1] = B(0);         /*   0     */
  a[2] = B(101);       /*   5     */
  a[3] = B(10101010);  /* 170     */

  ...
```

1. Gegeben ist das Programm `sort_permutation.c`, das eine Folge von Gleitkommazahlen einliest und anschließend die Zahlen aufsteigend sortiert ausgibt. Das Sortieren erfolgt mit Hilfe einer Sortierpermutation, also einem Feld mit Indizes, dessen Elemente anstelle der Gleitkommazahlen vertauscht werden.

   Ersetzen Sie in diesem Programm das Feld mit den Indexwerten durch ein Feld mit Zeigerwerten. Der Zugriff auf die Gleitkommawerte des zu sortierenden Feldes soll also nicht indirekt indiziert, sondern über Zeiger auf diese Feldelemente erfolgen.

2. Schreiben Sie ein Programm, das mehrere **float**-Werte einliest und zu jedem eingelesenen Wert die interne Darstellung ausgibt. Zur Ermittlung der Bitdarstellung ist der in der Übung „*Binärdarstellung von Integer-Zahlen*“ erstellte Algorithmus (als Funktion) zu verwenden.