1 String-Konkatenation

Implementieren Sie `string_concat` im folgenden Programm so, dass die beiden übergebenen Strings `string1` und `string2` zusammenhängend in den String `out` geschrieben werden. Benutzen Sie für die eigentliche Konkatenation keine von der Standardbibliothek bereitgestellten Funktionen wie z. B. `sprintf`. Testen Sie auch Spezialfälle wie leere Strings.

Erinnerung: Ein String ist ein Array von `char`-Werten, das mit einem Null-Byte (`\0`) terminiert wird.

```c
#include <stdio.h>

void string_concat(char out[], char string1[], char string2[])
{
}

int main(void)
{
  char out[128] = {'\0'};
  string_concat(out, "Hello", "World");
  printf("%s\n", out);
  return 0;
}
```

2 Struct, Unio, Enum

Modifizieren Sie das folgende Programm so, dass auf oberster Ebene nur noch ein `struct foobar` deklariert wird, das die beiden Strukturen `struct foo` und `struct bar` enthält. Um den Speicherplatz gering zu halten, soll außerdem eine Union verwendet werden, so dass `struct foobar` maximal so groß wie einer der beiden structs ist.

```c
#include <stdio.h>

struct foo
{
  int a;
  int b;
};

struct bar
{
  float a;
  float b;
};

int main(void)
{
  struct foo a = {42, 23};
  printf("foo: %d %d\n", a.a, a.b);
  struct bar b = {23.0, 42.0};
  printf("bar: %f %f\n", b.a, b.b);
  return 0;
}
```

Frage: Was passiert, wenn Sie trotz der Verwendung einer Union beide Structs beschreiben?

2.1 Enums

Passen Sie das gegebene Programm weiterhin so an, dass ein `enum` genutzt wird, um in `struct foobar` zu speichern, ob die Integer- oder die Float-Komponente aktiv ist. Die Größe von `struct foobar` darf sich dadurch erhöhen. Erweitern sie die main-Funktion um eine entsprechende beispielhafte Verwendung.

3 Pointer

Vollziehen Sie nach, was im folgenden Programm passiert. Passen Sie die print-Anweisung an, um Werte von a, b sowie die Werte, auf die pa, pb und ppb letztlich zeigen, aus.

```c
#include <stdio.h>

int main(void)
{
  int a = 10;
  int b = 20;

  int *pa = &a;
  int *pb = &b;
  int **ppb = &pb;

  *pa = *pa + 5;
  pa = pb;
  *pa = *pa + 5;

  printf("TODO");

  return 0;
}
```

Material

Makefile und Code-Vorlagen liegen dem beiliegenden Übungsmaterial bei. Sie können alle Binaries automatisch mittels `make` oder gezielt mit `make BINARYNAME` bauen lassen.