# Serie 2

Computer Architecture 2023
University of Bern

14.03.2023

The assignment consists of **theoretical** and **programming** parts:

- the **correct answer** to a theoretical question **without explanation** will be **evaluated with 0 points**
- **make sure that your code** for the programming part **compiles without any errors** or warnings
- **plagiarism is not allowed!**

**Upload** your solutions:

- a ***pdf*** file with answers to theoretical questions
- a ***zip* archive** with all necessary *.c* and *.h* files to run your code

## Theoretical Part

### 1. Single-functional *for* Loop [3 points]

Is this loop infinite and what will be printed by this code snippet? Explain your reasoning.

```c
#include <stdio.h>


int get_number() {
    static int number = 8;
    return --number;
}


int main() {
    for (get_number(); get_number(); get_number()) {
        printf("%d ", get_number());
    }

    return 0;
}
```

## 2. Asterisk and Pointifix: Mission Dereference [3 points]

What will be printed by this code snippet? Explain your reasoning.

```c
#include <stdio.h>


int main() {
    int arr[2][2][2] = {{{1, 2},
                         {3, 4}},
                        {{5, 6},
                         {7, 8}}};

    int x = *(**arr + 1);
    int y = *(*(*arr + 1) + 1);
    int z = **(*(arr + 1) + 1);

    printf("%d %d %d", x, y, z);

    return 0;
}
```

## 3. Asterisk and Pointifix *vs.* Incrementor [3 points]

What will be printed by this code snippet? Explain your reasoning.

```c
#include <stdio.h>


int main() {
    int arr[2][2][2] = {{{1, 2},
                         {3, 4}},
                        {{5, 6},
                         {7, 8}}};

    int (*p)[2][2] = arr;

    int x = *(**++p + 1);
    int y = *(*(*p--) + 1);
    int z = ***p;

    printf("%d %d %d", x, y, z);

    return 0;
}
```

## 4. A Short String Break From Pointers [1 point]

What will be printed by this code snippet? Explain your reasoning.

```c
#include <stdio.h>


int main() {
    char phrase[] = "hello";
    char *p = phrase;

    printf("%s", p + p[0] - p[1]);

    return 0;
}
```

## 5. Pointers are Everywhere [2 points]

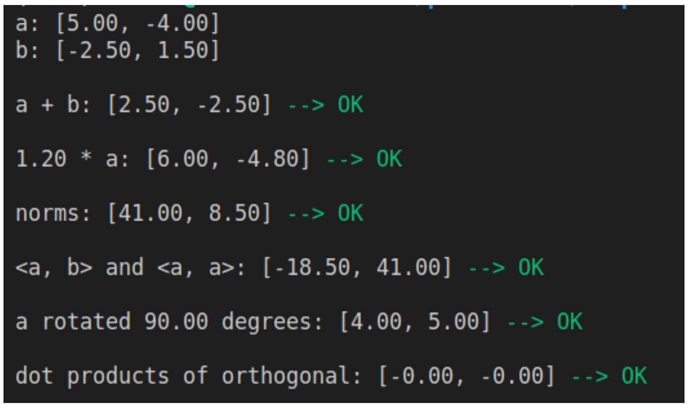
What will be printed by this code snippet? Explain your reasoning.

```c
#include <stdio.h>


int add(int a, int b) {
    return a + b;
}

int multiply(int a, int b) {
    return a * b;
}


int main() {
    int (*function[])(int, int) = {add, multiply};
    int (*p)(int, int) = *function;

    printf("%d ", (*(p++))(2, 3));
    printf("%d", (*(--p))(2, 3));

    return 0;
}
```

## Programming Part

In this task, the goal is to create operations on 2D vectors. You should write code in all incomplete functions inside files *vector.h* and *vector_printing.h*. **You are not allowed to change anything in the *test.c* file!**

Run the *test.c* to be sure that everything is working.
After running *test.c* you should get the following (in case of successful implementation):



```
a: [5.00, -4.00]
b: [-2.50, 1.50]

a + b: [2.50, -2.50] --> OK

1.20 * a: [6.00, -4.80] --> OK

norms: [41.00, 8.50] --> OK

<a, b> and <a, a>: [-18.50, 41.00] --> OK

a rotated 90.00 degrees: [4.00, 5.00] --> OK

dot products of orthogonal: [-0.00, -0.00] --> OK
```

To avoid problems with *math.h* library (due to using *sin*, *cos* functions) you should use the command line (terminal) to compile and run the *test.c* file:

```
gcc test.c -o test -lm
./test
```