Programming in C I
Jacobs University Bremen
Dr. Kinga Lipskoch
Instructor: Xu He

Course: JTSK-320111
Fall 2017

# Grading Criteria

## 1 Your TAs

- Alexandru Sasu (a.sasu@jacobs-university.de)
- Leul Abiy Shiferaw (l.shiferaw@jacobs-university.de)

## 2 Code formatting

- Indentation is a way to organize the document, make sure you indent your code. This will make your work easier to read/understand/modify/maintain/debug by anyone who sees your code or even by you after a long time.
- You can use `i`, `j`, `k` variables as iterators.

| **Deduction for NOT following the criterion** | **What do we expect from you** |
|---|---|
| $-5\%$ | There are many styles of indenting your code (in C the most common ones are K&R and Allman). It is your choice which one to use, however we ask that all your programs will be indented only in the one chosen way |
| $-5\%$ | Name the variables according to their meaning (e.g., `numberOfCars` and not `x`) |
| $-5\%$ | Use at most 80 characters per line to improve readability of your code in JGrader |
| $-5\%$ | Use only the English alphabet (i.e., stick to ASCII characters) |

### 2.1 Examples of code formatting

- K&R style

```c
#include <stdio.h>

int main(){
    int numberOfCars = 1, index = 0;

    while(index < 3){
        numberOfCars *= 15;
        index++;
    }

    printf("%d\n", numberOfCars);
    return 0;
}
```

- Allman style

```c
#include <stdio.h>

int main()
{
    int numberOfCars = 1, index = 0;

    while(index < 3)
    {
        numberOfCars *= 15;
        index++;
    }

    printf("%d\n", numberOfCars);
    return 0;
}
```

# 3 Comments

| Deduction for NOT following the criterion | What do we look for |
|---|---|
| −5% | Comment out the important parts of your code, be as precise as possible and avoid commenting every line. Comments should describe your code: no questions, suggestions, or complaints should be included. Put in the beginning of your program a block of comments (look for example below). |

## 3.1 Example of the header block for your program

```
/*
    JTSK-320111
    problem 1.1.c
    Student Name
    j.email@jacobs-university.de
*/
```

**Good example:**

```
2
3   int cars[100], length = 0;
4
5       for (int i = 0; i < length - 1; i++)
6           for (int j = 0; j < length; j++)
7               if (cars[i] > cars[j])
8                   // sort array of cars in ascending order
9                   swap(cars[i], cars[j]);
10      return 0;
11  }
```

**Bad example:**

```
6
7       // array of cars of size 100, variable length of 0 value
8       int cars[100], length = 0;
9       // for loop to go from 0 to length - 1
10      for (int i = 0; i < length - 1; i++)
11          // another for loop which goes from 0 to length
12          for (int j = 0; j < length; j++)
13              // compare cars[i] and cars[j]
14              if (cars[i] > cars[j])
15                  // swap cars[i] and cars[j]
16                  swap(cars[i], cars[j]);
17      return 0;
18  }
19
```

# 4 Compiling

The code should compile with no errors or warning messages. Otherwise TA will use the following table for deduction:

| Maximum points subtracted | Criterion |
|---|---|
| −10% | Program compiles with warnings |
| −50% | Program does not compile |
| −30% | Segmentation fault |

# 5 Correctness and completeness

| **Deduction for NOT meeting the expectation** | **What do we expect from you** |
|---|---|
| $-5\%$ | Check for null pointers |
| $-20\%$ | Allocate and deallocate memory correctly |
| $-35\%$ | Do not make logical mistakes in functions and be sure your functions behave as they are meant to |
| $-35\%$ | Follow the guidelines for the given problem and make sure you meet all requirements |
| $-20\%$ | Check whether your program works for all cases (please check edge cases) |

We would like to see your own work and assess your own effort in completing the tasks for the course. If we find two completely identical or very similar codes we will penalized students with $100\%$ deduction for the task.