**Exercise 2**  
C/C++ “Warming up II”  
(handed out: 27.04.2020 – hand in: 04.05.2020)

**Review**

1. **What must be in every C/C++ program (1P)?**
   - `#include`
   - `main() {}`
   - `return`

2. **How to loop correctly over the whole array `int m[n]` (1P)?**
   - `for (int i=1; i<=n; i++)`
   - `for (int i=0; i<n; i++)`
   - `for (int i=1; i<n; i++)`
   - `for (int i=0; i<=n; i++)`

---

1. **Task** Floating point number representation II (1 P)
   Write a C/C++ program that performs the following inversion test for $x = 0.1, \ldots, 1000$ in steps of 0.1:
   
   ```
   y = 1. / x ;
   z = y * x ;
   if ( z != 1. ) k++ ;
   ```

   How often does the inversion test fail?

2. **Task** Radius calculation (3 P)
   Write a C++ program for calculating the radius of a star from its given luminosity $L_*$ and its given effective temperature $T_{\text{eff}}$ as in
   
   \[ L_* = 4 \pi \sigma_{\text{SB}} R_*^2 T_{\text{eff}}^4. \]  
   (1)

   The user should be asked for $\log(L_*/L_\odot)$ (luminosity in units of solar luminosity) and $T_{\text{eff}}/K$ and obtains in return the stellar radius in $R_\odot$ (solar radii).

   **Hint:** You don’t need the value of $\sigma_{\text{SB}}$, but only the effective temperature of the sun: $T_{\text{eff}} = 5778$ K. (2 P) What is the numerical advantage of doing so? (1 P)
   
   E.g., which radius has a white dwarf of $T_{\text{eff}} = 144$ kK and $\log(L_*/L_\odot) = 3.8$? (Answer: $R = 0.13 R_\odot$).

   **Additional task:** Try to catch invalid input, e.g., negative temperatures (easy, 1 extra P) or letters (difficult, 2 extra P).

3. **Task** Importing data into arrays (4 P)
   Write a C++ program that imports x-y data pairs into an array.
   
   a) First, arrays $x[100]$ and $y[100]$ of type `double` shall be declared. (0.5 P)
   b) The user should specify how many data pairs $n$ he wants to enter. (0.5 P)
c) The program then imports via a for-loop the x- and y-values entered by the user into the arrays x[] and y[] (0.5 P):
    ```cpp
    cin >> x[i] >> y[i] ;
    ```
d) The imported data should be printed out pairwise in form of an x-y table for checking. (0.5 P)

The imported data points shall be analyzed (2 P):
e) The line of best fit for the entered values shall be determined:

\[
    y = b \cdot x + a
\]  

where

\[
    b = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}, \tag{3}
\]

\[
    a = \bar{y} - b \cdot \bar{x}, \tag{4}
\]

with the means \( \bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \) and \( \bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i \). The coefficients \( a, b \) shall be printed out.

4. Task Pointers, references, arrays (6 P)

As we will use pointers, references, and arrays in upcoming exercises it might be helpful to bring their functionality to mind:

a) Write a short C++ program which uses references as shown in the lecture (subsection “References”).

Pointers and arrays correspond to each other in some way. Have a look on the following program lines and answer the questions:

```cpp
    int array[4] ; // (1)
    int *parray = 0 ; // (2)
    parray = array ; // (3)
    parray[3] = 1 ; // (4)
    parray = &array[2] ; // (5)
    parray[1] = 6 ; // (6)
```

b) Why should a pointer always be initialized with 0, as in line (2)?

c) By the intriguing assignment in line (3) the pointer contains now the start address of the array. How is this assignment normally done?

d) The pointer behaves then like an array, i.e. it can be “indexed” by square brackets. How does this work?

e) What is the effect of the lines (5) and (6)?

f) At which array index is then the value 6 stored?

(each sub task: 1P)