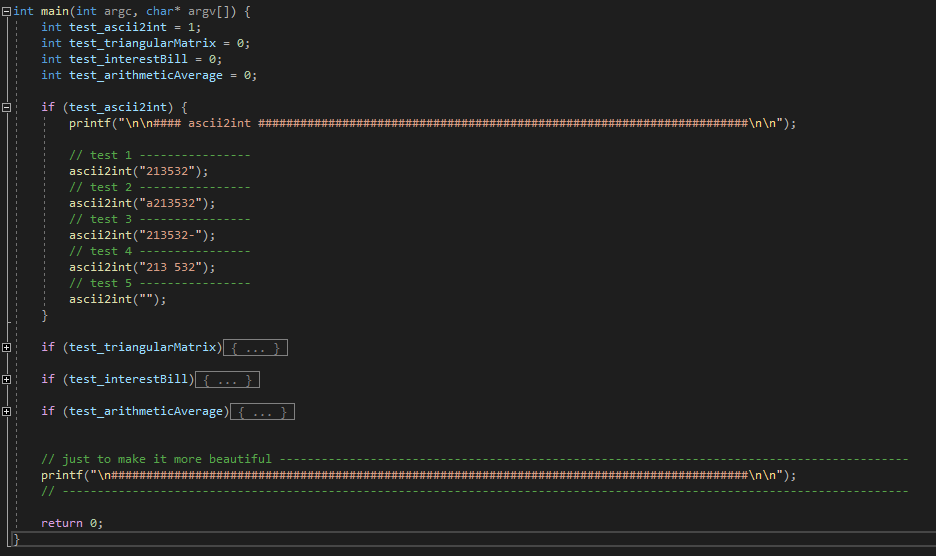
1 ascii2int

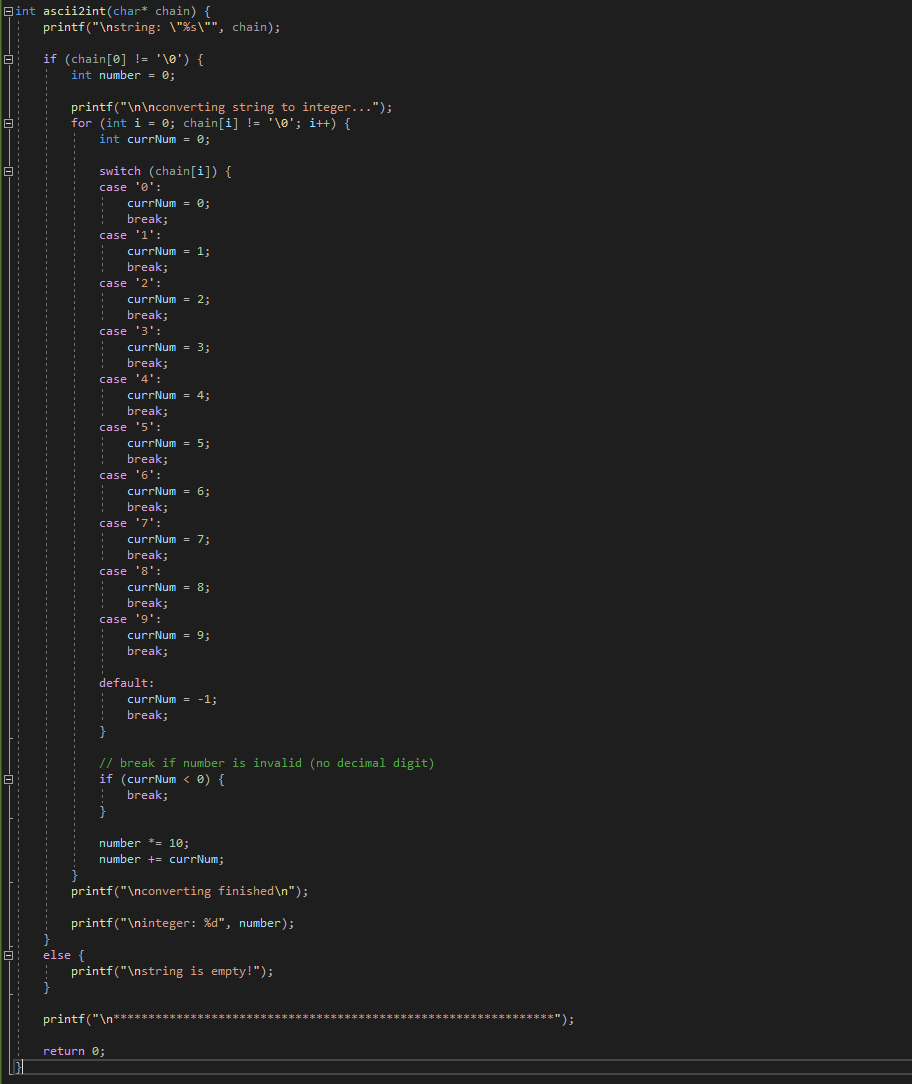
Lösungsidee:  
Zuerst werden die Übergabeparamter überprüft. Wenn alle valid sind, wird zuerst eine integer (number) Variable deklariert und dann mit einer Schleife über das character array drüber iteriert und mit ‚switch case‘ bei jeder Dezimalziffer die entsprechende Nummer in eine Variable gespeichert, die dann anschließend mit 10 multiplieziert und dann zur Variable ‚number‘ addiert wird. Falls in der Schleife einen nicht dezimalen Character erreicht wird, wird aus der Schleife gesprungen.   
  
Testfälle: „213532“, „a213532“, „213532-“, „213 532“ und „“

C-Program code:

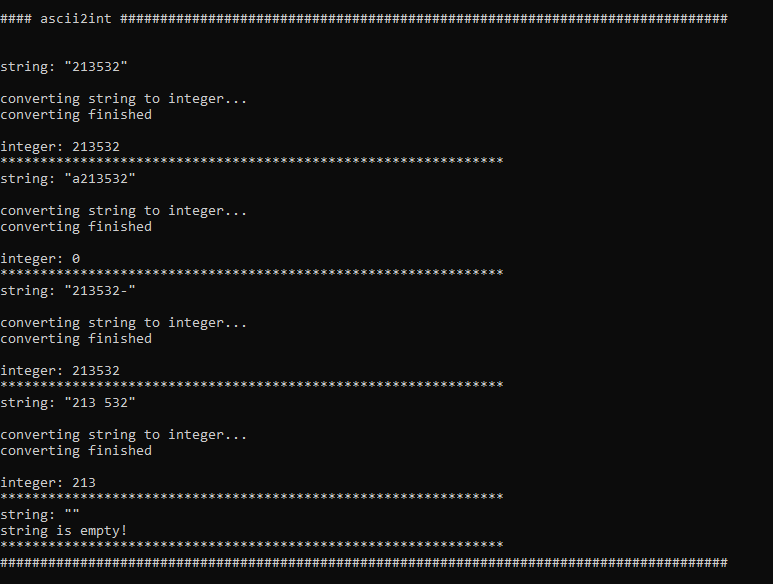
Int main:



Int ascii2int:



Test Ausgaben:   
(Start values -> „213532“, „a213532“, „213532-“, „213 532“ und „“)



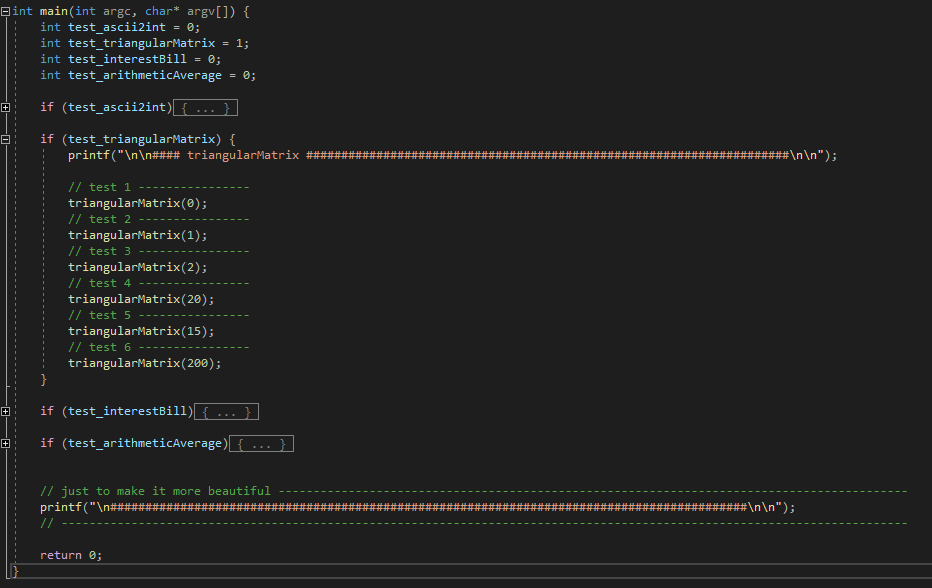
2 Dreiecksmatrix

Lösungsidee:  
Zuerts werden die Übergabeparameter überprüft. Wenn alle valid sind wird in der Schleife jeder ungerade Wert wie eine linksbündige Pyramide ausgeben. Dabei fängt die Pyramide mit einem Wert an und wird bei jeder Stufe um eins erhöt. Um dies zu erreichen werden drei Variabeln benötigt. Eine die den ‚line break‘ bestimmt, eine die die ausgegebenen Nummern in der aktuellen Reihe zählt und eine die die ungerade Nummer besitzt. Wenn die ‚nummer zähler‘ Variable modulo mit der ‚line break‘ Variable gerechnet 0 ergibt wird ein ‚Line break‘ ausgeführt.

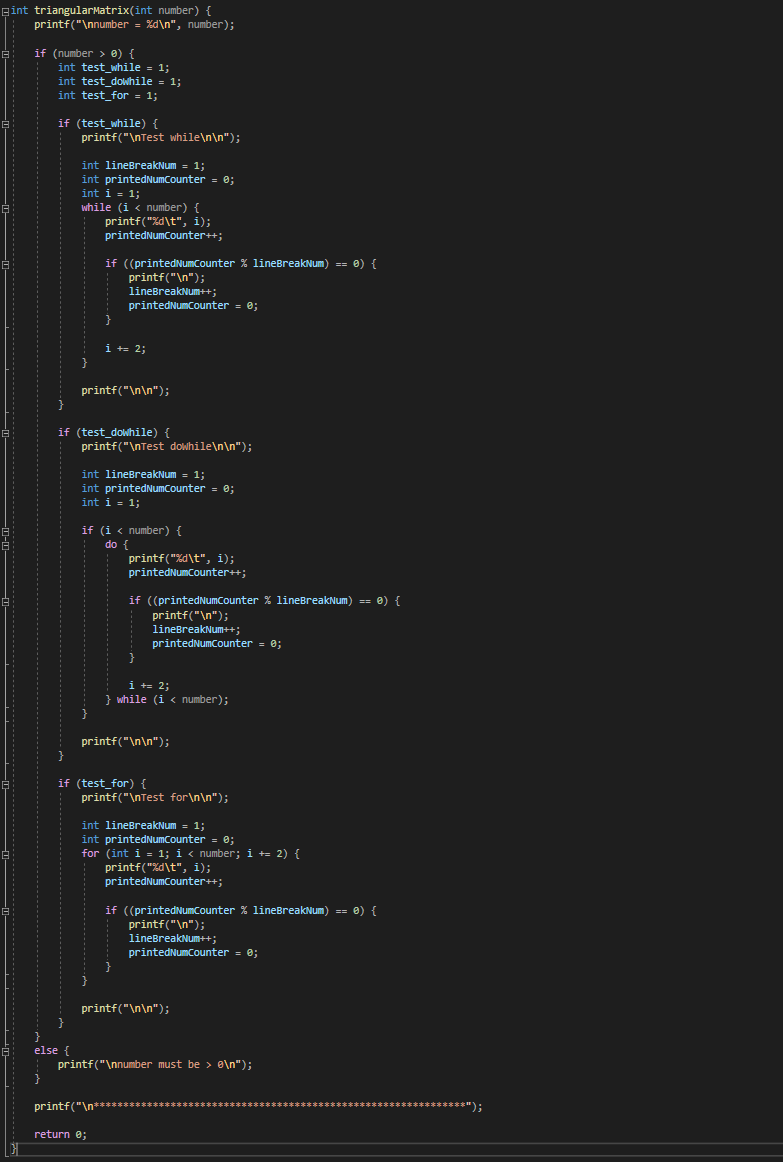
Testfälle: 0, 1, 2, 20, 15, 200

C-Program code:

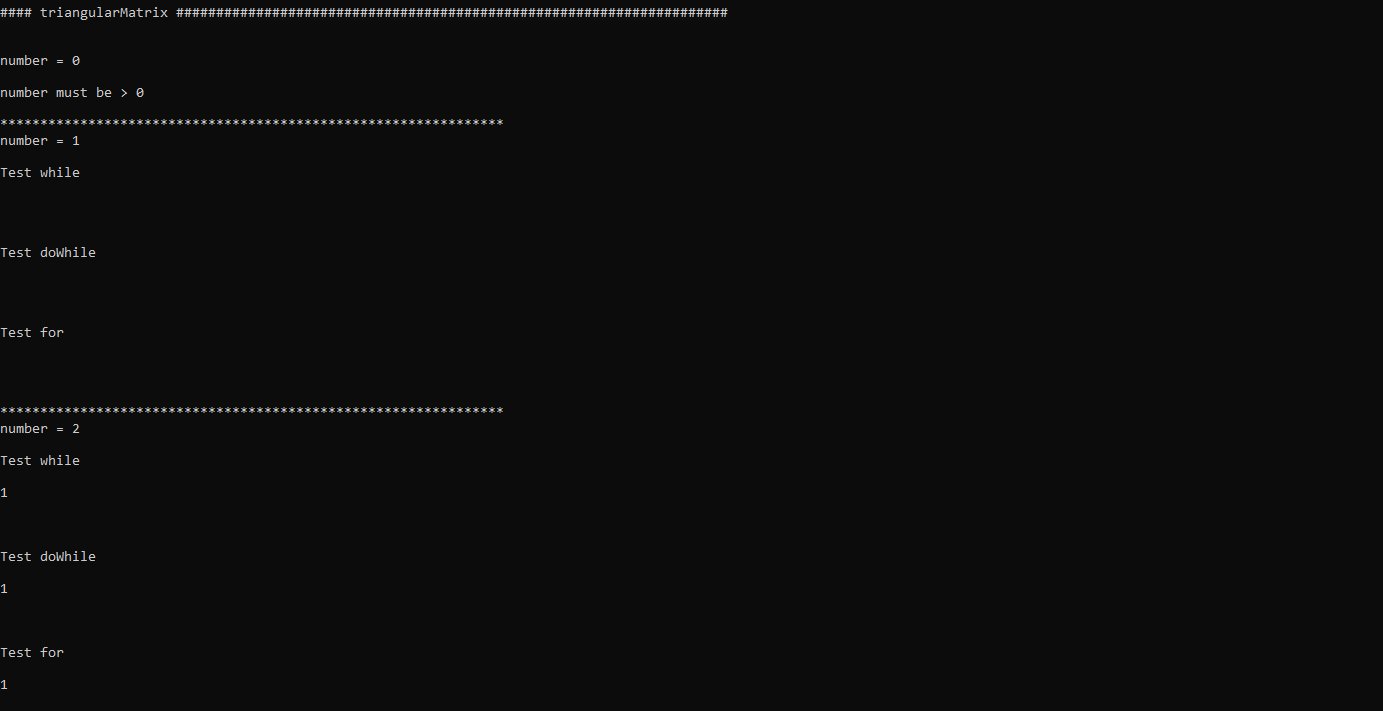
Int main:

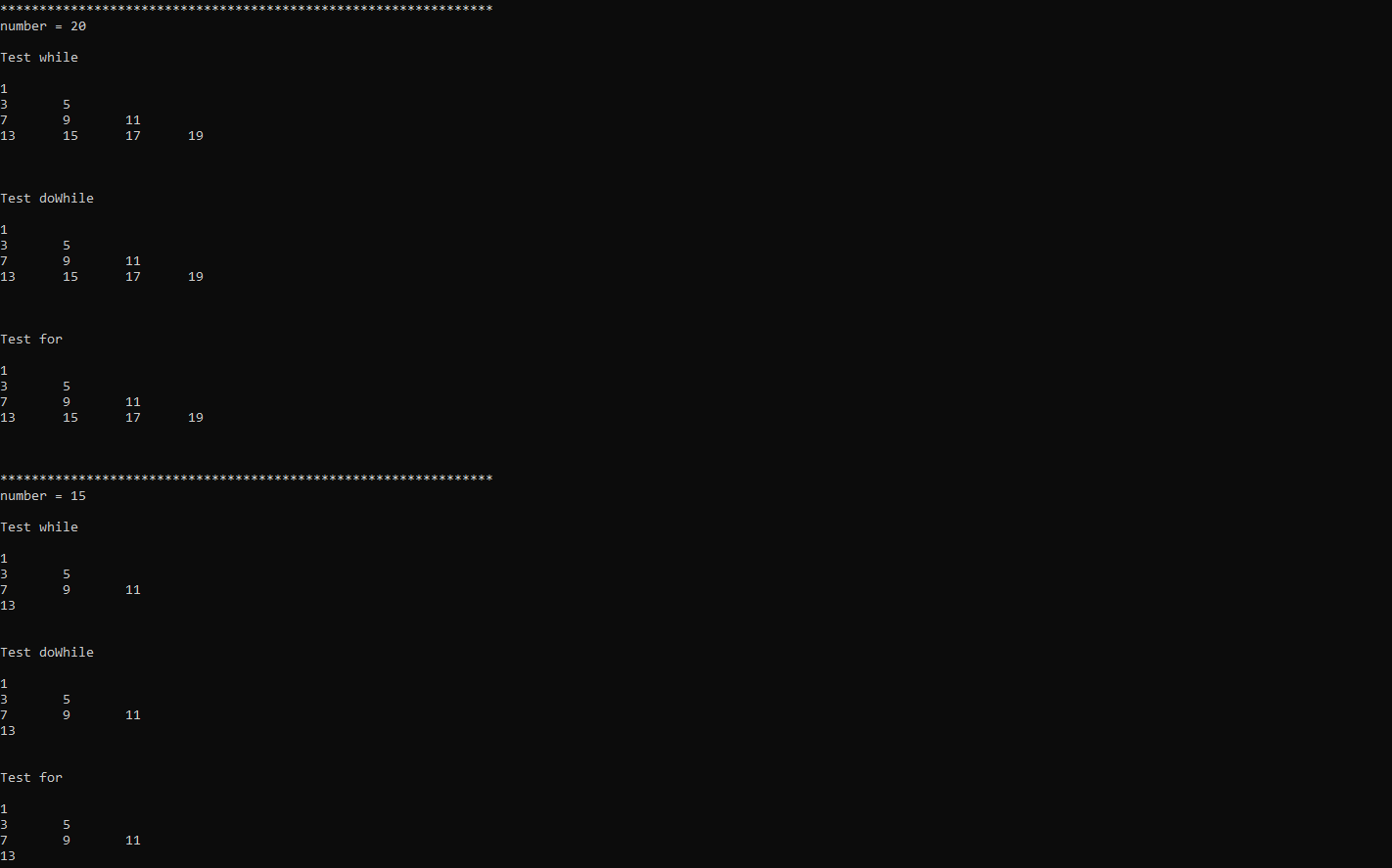


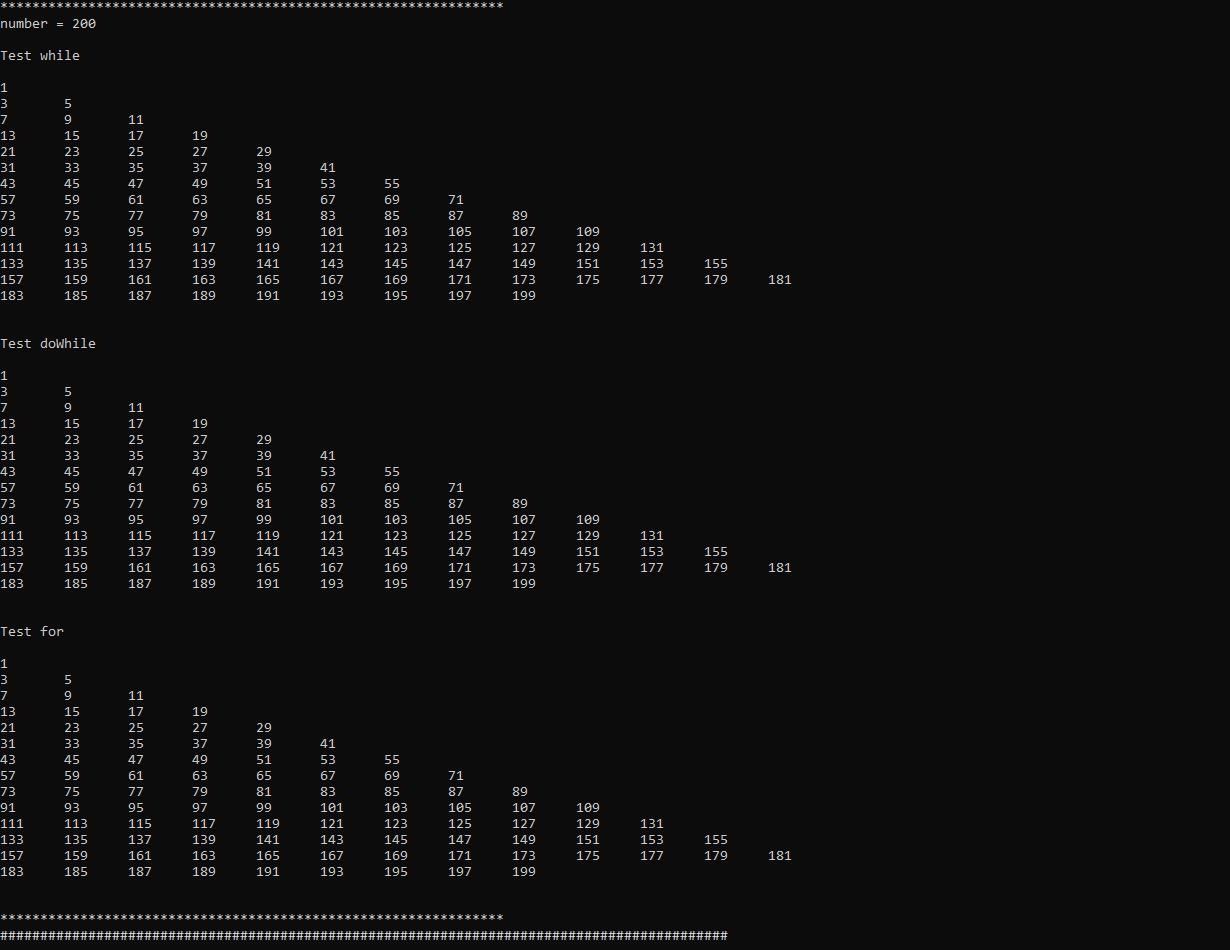
Int triangularMatrix:



Test Ausgaben:   
(Start values -> „213532“, „a213532“, „213532-“, „213 532“ und „“)







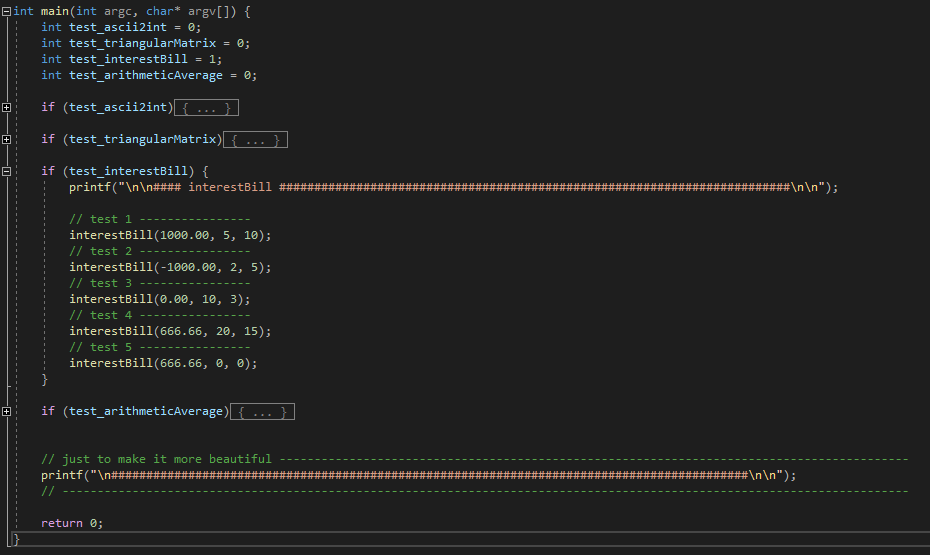
3 Zinsrechnung

Lösungsidee:  
Zuerts werden die Übergabeparameter überprüft. Wenn alle valid sind wird, je nach dem wie viele Jahre man angegeben hat, jedes Jahr zum Startkapital der entsprechende Zinssatz hinzugefüght.  
Am Ende wird dann das Endkapital ausgegeben.

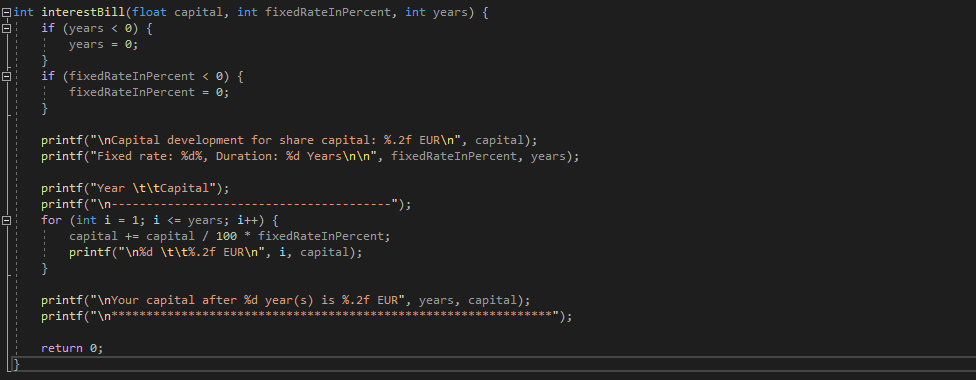
Testfälle: (1000.00, 5, 10) (-1000.00, 2, 5) (0.00, 10, 3) (666.66, 20, 15) (666.66, 0, 0)

C-Program code:

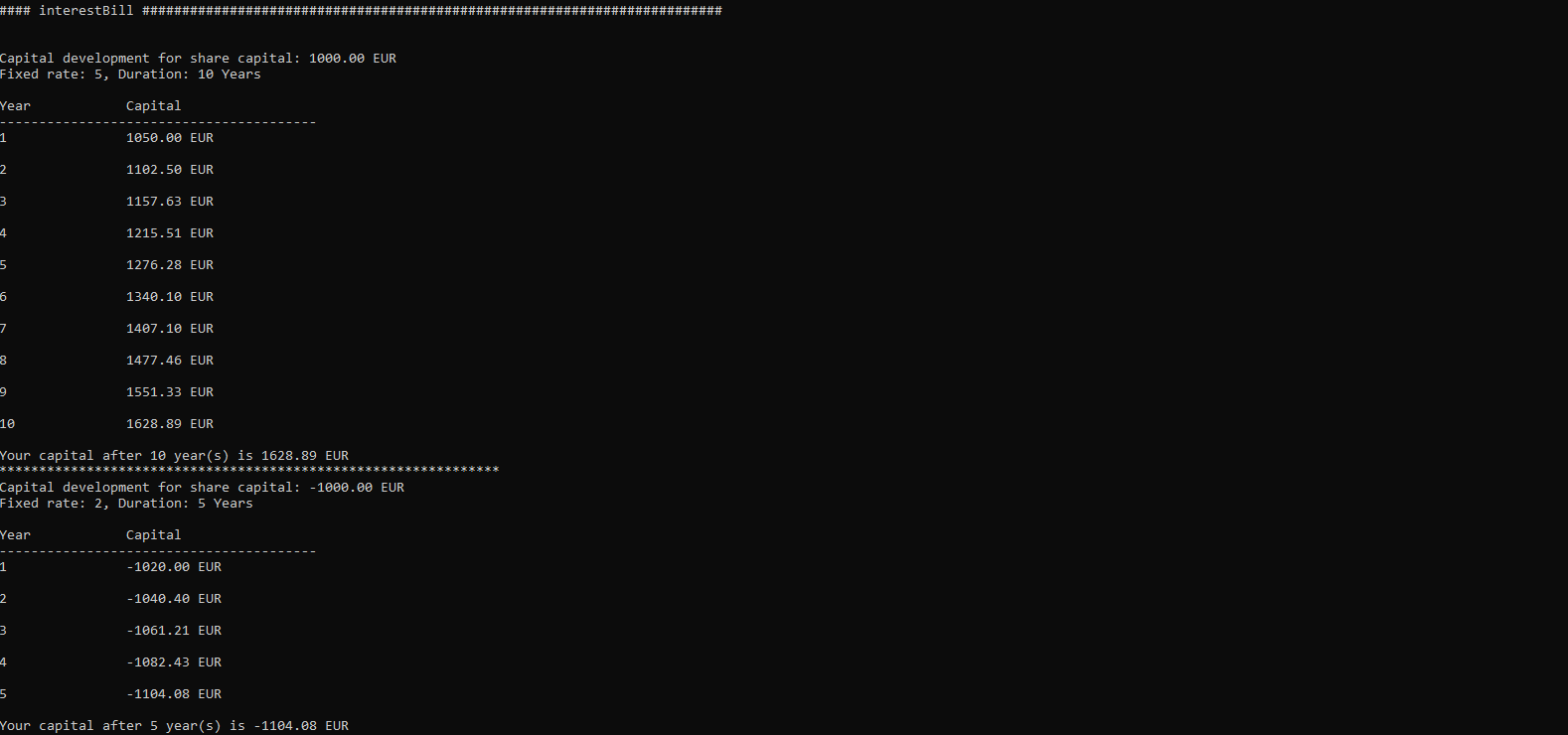
Int main:

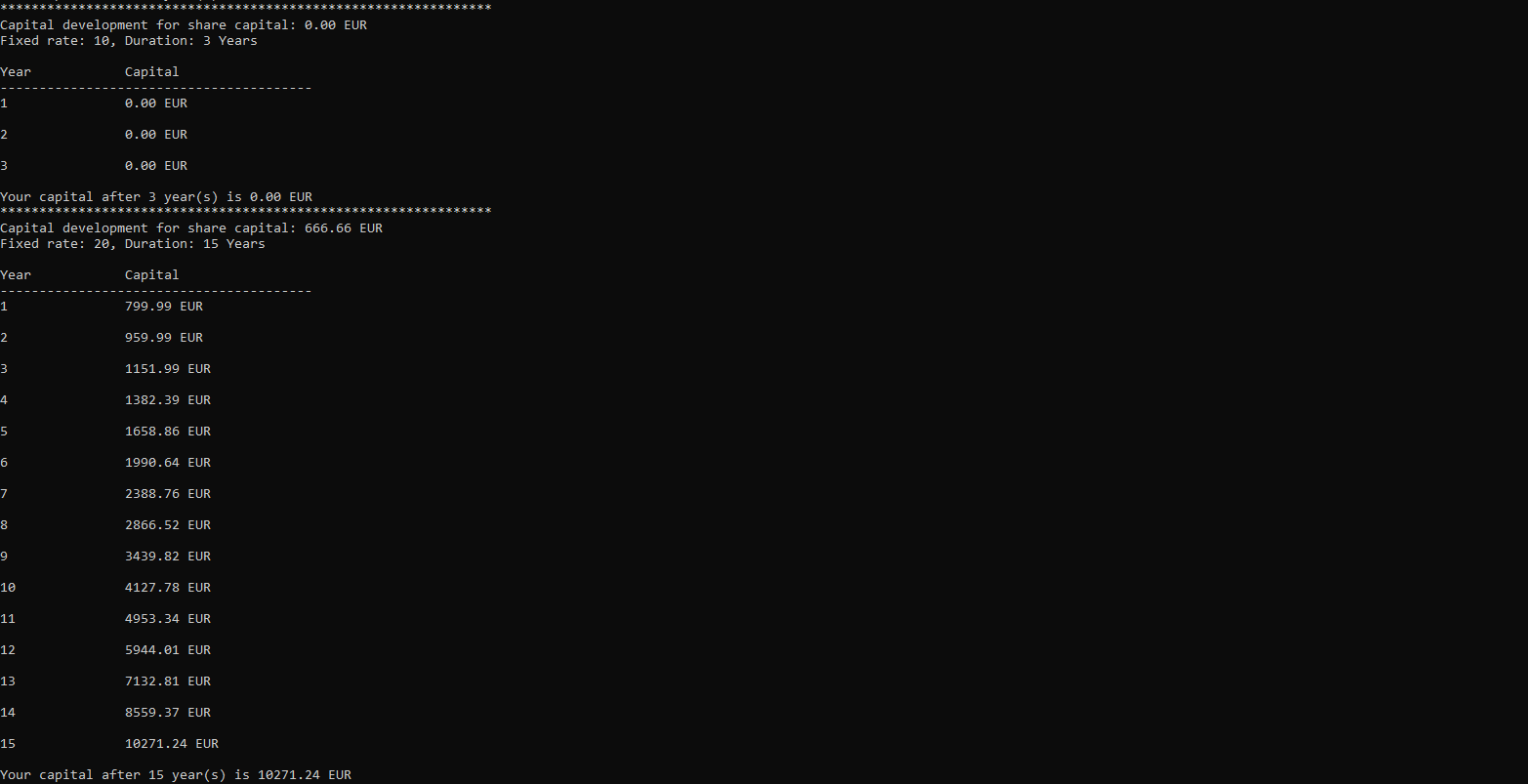


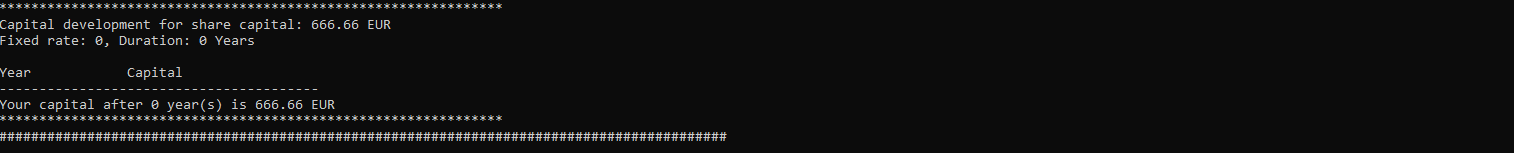
Int interestBill:



Test Ausgaben:   
(Start values -> (1000.00, 5, 10) (-1000.00, 2, 5) (0.00, 10, 3) (666.66, 20, 15) (666.66, 0, 0))







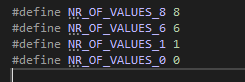
4 Arithmetisches Mittel + Ausreißer

Lösungsidee:  
Zuerts werden die Übergabeparameter überprüft. Die länge der Arrays wird mit ‚#define‘ definiert. Die min/max Variabeln werden mit dem ersten Wert vom array intialisiert damit die min/max Variabeln einen richtigen Anfangswert haben. Dannach wird mit der vordefinierten länge vom Array durch das Array iteriert und in jeder Iteration wird überprüft ob es ein neuen min/max Wert gibt. Am Ende der Schleife wird der aktuelle Wert in einer von außerhalb der Schleife deklarierte Variable hinzugezählt.

Testfälle: { 3, 5, 7, 8, 1, -1, 4, 0 } { 0, 4, 1, -10, 4, 0 } { 2 } { }

C-Program code:

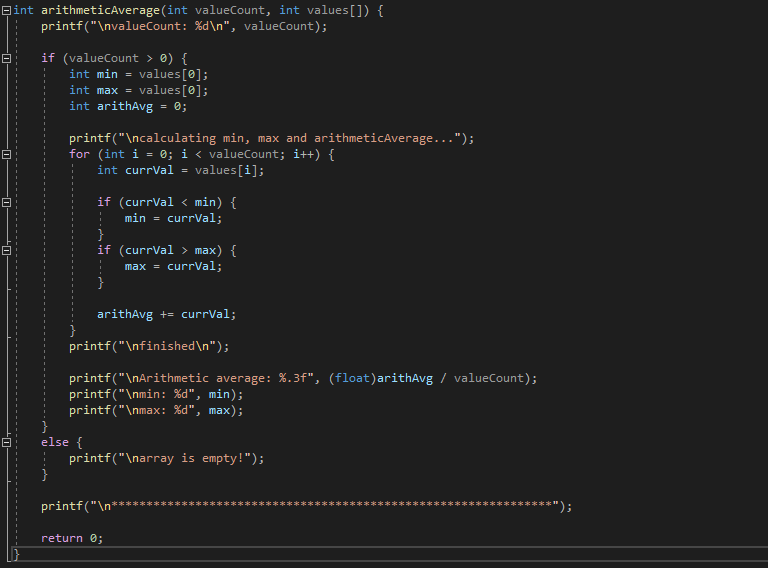
#defines:



Int main



Int arithmeticAverage:



Test Ausgaben:   
(Start values -> { 3, 5, 7, 8, 1, -1, 4, 0 } { 0, 4, 1, -10, 4, 0 } { 2 } { })

