```
+--------------------------------------------------------------+
| ULAM (Blatt 01)
+--------------------------------------------------------------+

Ulrike Griefahn
Tue Jan  3 21:25:35 CET 2017

+--------------------------------------------------------------+
| Eingesendete Dateien                                         |
+--------------------------------------------------------------+

ulam.c
```

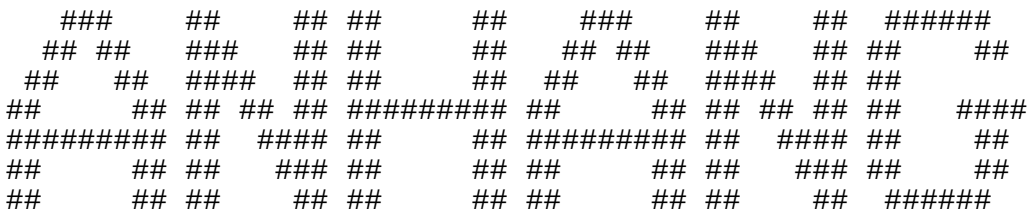
```
+-------------------------------------------------------------+
| Loesung mit Splint pruefen                                  |
+-------------------------------------------------------------+

Splint 3.1.2 --- 03 May 2009

Finished checking --- no warnings
```

```
+------------------------------------------------------------+
| Programmkommentare mit Doxygen pruefen                     |
| Doxygen 1.8.6
+------------------------------------------------------------+

warning: The selected output language "german" has not been updated
since release 1.8.4.  As a result some sentences may appear in English.

Doxygen done
```

```
+-------------------------------------------------------------+
| Kompilieren mit C89-Standard                                |
+-------------------------------------------------------------+

Compilation done
```

```
+------------------------------------------------------------+
| Kompilieren mit C99-Standard                               |
+------------------------------------------------------------+

Compilation done
```

```
+------------------------------------------------------------+
| TEST ULAM (Blatt 01)
+------------------------------------------------------------+

=========================================================
Ueberpruefe Testfaelle ( U L A M _ M A X ):
=========================================================
Testfall 1 ulam_max: Ungueltiger Wert fuer Parameter a_0
[OK]

[OK]

Testfall 2 ulam_max: Gueltige Werte fuer Parameter a_0
[OK]

[OK]

[OK]

[OK]

[OK]

=========================================================
Ueberpruefe Testfaelle ( U L A M _ T W I N S ):
=========================================================
Testfall 3 ulam_twins: Ungueltiger Wert fuer Parameter limit
[OK]

Testfall 4 ulam_twins: ULAM-Zwilling nicht vollstaendig im Intervall enthalten
[OK]

Testfall 5 ulam_twins: Gueltige Werte fuer Parameter limit
[OK]

=========================================================
Ueberpruefe Testfaelle ( U L A M _ M U L T I P L E S ):
=========================================================
Testfall 6 ulam_multiples: Ungueltiger Wert fuer Parameter limit
[OK]

Testfall 7 ulam_multiples: Ungueltiger Wert fuer Parameter number
[OK]

Testfall 8 ulam_multiples: Mehrling fuer number=2 nicht vollstaendig im Interval
l enthalten
[OK]

Testfall 9 ulam_multiples: Mehrling fuer number=4 nicht vollstaendig im Interval
l enthalten
[OK]

Testfall 10 ulam_multiples: Gueltige Werte fuer Parameter limit und number
[OK]

[OK]

[OK]

[OK]


+------------------------------------------------------------+
| Tests (total): 18, Tests (passed): 18, Rating: 100 %       |
+------------------------------------------------------------+
```



```
+------------------------------------------------------------------------+
| Listing der Datei:                                                     |
| ./blatt01/submissions/Ulrike Griefahn_3047252/ulam.c
+------------------------------------------------------------------------+

/**
 * @mainpage
 *
 * <h1>Projekt ULAM</h1>
 *
 * Dieses Projekt implementiert Berechnungen mit der ULAM-Funktion. Es werden
 * ULAM-Folgen, ULAM-Zwillinge und ULAM-Mehrlinge berechnet.
 *
 * @author  Ulrike Griefahn
 * @date    2014-10-05
 */

/* ======================================================================*/

/**
 * @file
 * Dieses Modul implementiert Berechnungen mit der ULAM-Funktion. Es werden
 * ULAM-Folgen, ULAM-Zwillinge und ULAM-Mehrlinge berechnet.
 *
 * @author  Ulrike Griefahn
 * @date    2014-10-05
 */

/* ======================================================================
 * Header-Dateien
 * ====================================================================== */

#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <assert.h>



/* ======================================================================
 * Symbolische Konstanten
 * ====================================================================== */

/**
 * Erreicht ein ungerader Wert der ULAM-Folge den Wert #ULAM_MAX, kann die Folge
 * nicht weiter berechnet werden, da es sonst zu einem berlauf kommen wrde.
 */
#define ULAM_MAX INT_MAX / 3 + 1


/* ======================================================================
 * Funktions-Prototypen
 * ====================================================================== */

/**
 * Liefert den nchsten ULAM-Wert zu einer positiven ganzen Zahl. Die
 * ULAM-Funktion ist wie folgt definiert:
 *
 * ULAM(an) =
 * <ul>
 *   <li> an / 2,     falls an gerade und an > 1
 *   <li> 3 * an + 1, falls an ungerade und an > 0
 * </ul>
 *
 * Die Funktion liefert -1, wenn an <= 0 oder wenn es whrend der Berechnung
 * zu einem berlauf kommen wrde.
```

```
 *
 * @param an        positive ganze Zahl, zu der der nchste ULAM-Wert
 *                  geliefert werden soll.
 * @return          der nchste ULAM-Wert zur bergebenen Zahl
 */
int ulam(int an);

/**
 * Liefert fr eine positive ganze Zahl a0 den maximalen Wert in der Folge
 * ihrer ULAM-Werte, bspw. liefert ulam_max(5) den Wert 16 und ulam_max(7) den
 * Wert 52. Wird eine Zahl < 1 bergeben, liefert die Funktion den Wert -1.
 *
 * Die Funktion liefert -1, wenn a0 < 1 oder wenn es whrend der Berechnung
 * zu einem berlauf kommen wrde.
 *
 * @param a0        ganze Zahl, zu der der maximale ULAM-Wert geliefert
 *                  werden soll.
 * @return          der maximale ULAM-Wert zur bergebenen Zahl
 *                  oder -1 bei berlauf oder wenn a0 < 1 ist
 */
int ulam_max(int a0);

/**
 * Prft fr alle positiven ganzen Zahlen a0 von 1 bis einschlielich limit,
 * ob es ULAM-Zwillinge gibt, d.h. ob zwei benachbarte Werte a0 und a0+1 im
 * Intervall vollstndig enthalten sind, deren maximale ULAM_Werte gleich sind.
 *
 * Die Funktion liefert a0, wenn ein Paar gefunden wurde und a0 die kleinere
 * Zahl in einem solchen Paar ist. Sind mehrere Paare im Intervall enthalten,
 * wird der kleinere Index des letzten Paars zurck gegeben.
 *
 * Die Funktion liefert -1, wenn kein solches Zwillingspaar gefunden wird oder
 * es whrend der Berechnung zu einem berlauf kommen wrde..
 *
 * @param limit     ganze Zahl, bis zu der nach ULAM-Zwillingen
 *                  gesucht werden soll.
 * @return          die kleinere Zahl des letzten ULAM-Zwillingspaars
 *                  oder -1 bei berlauf oder wenn es kein solches Paar gibt
 *                  oder limit < 1 ist
 */
int ulam_twins(int limit);

/**
 * Prft, ob im Intervall von 1 bis einschlielich limit ULAM-Mehrlinge
 * mit der Anzahl number vollstndig enthalten sind, d.h. fr number = 3
 * Drillinge, fr number = 4 Vierlinge usw.
 *
 * Die Funktion liefert a0, wenn Mehrlinge gefunden wurden und a0 die
 * kleinste Zahl ist, die zu den Mehrlingen gehrt. Sind weitere
 * Mehrlingsgruppen im Intervall enthalten, wird der kleinste Index der
 * letzte Gruppe zurck gegeben.
 *
 * Die Funktion liefert -1, wenn die Parameter nicht sinnvoll sind
 * (limit < number, number < 2), keine solchen Mehrlinge gefunden wurden
 * oder es whrend der Berechnung zu einem berlauf kommen wrde.
 *
 * Bspw. liefert ulam_multiples fr (1000, 2) den Wert 5, fr (1000, 3) den
 * Wert 107, fr (108, 3) den Wert -1 und fr (391, 6) den Wert 386.
 *
 * @param limit     positive ganze Zahl, bis zu der nach ULAM-Mehrlingen
 *                  gesucht werden soll.
 * @param number    positive Zahl, die die Anzahl der gesuchten Mehrlinge angibt
 * @return          die kleinste Zahl des letzten "ULAM-Mehrlings" oder -1, wenn
 *                  keine Mehrlinge gefunden wurden, limit < number oder
 *                  number < 2.
 */
```

```
int ulam_multiples(int limit, int number);


/* ========================================================================
 * Funktionsdefinitionen
 * ======================================================================== */

/* ------------------------------------------------------------------------
 * Funktion: ulam
 * ------------------------------------------------------------------------ */
int ulam(int an)
{
    int ulam_an;

    /* Fr negative Zahlen und 0 gibt es keinen nchsten ULAM-Wert. */
    if (an < 1)
    {
        return -1;
    }

    /* Berechnung des nchsten ULAM-Werts */
    if (an % 2 == 0)
    {
        /* an ist gerade */
        ulam_an = (an / 2);
    }
    else
    {
        /* an ist ungerade */
        if (an < ULAM_MAX)
        {
            ulam_an = (3 * an + 1);
        }
        else
        {
            printf("Ueberlauf bei Berechnung des Ulam-Wert von %d\n", an);
            ulam_an = -1;
        }
    }

    return ulam_an;
}

/* ------------------------------------------------------------------------
 * Funktion: ulam_max
 * ------------------------------------------------------------------------ */
int ulam_max(int a0)
{
    int an;             /* Zahl, deren ULAM-Wert berechnet wird */
    int ulam_value;     /* ULAM-Wert zu an */
    int max_ulam_value; /* max. ULAM-Wert in der Folge von a0 bis an */

    /* Fr negative Zahlen und 0 kann kein maximaler ULAM-Wert berechnet
     * werden. */
    if (a0 < 1)
    {
        return -1;
    }

    /* Berechnung des maximalen ULAM-Werts fr a0 */
    an = a0;
    max_ulam_value = a0;

    while (an > 1)
    {
        ulam_value = ulam(an);
```

```
        if (ulam_value > max_ulam_value)
        {
            max_ulam_value = ulam_value;
        }
        an = ulam_value;
    }

    return max_ulam_value;
}

/* ------------------------------------------------------------------------
 * Funktion: ulam_twins
 * ------------------------------------------------------------------------ */
int ulam_twins(int limit)
{
    int ulam_max_1;     /* ULAM-Max-Werte von zwei benachbarten Zahlen */
    int ulam_max_2;
    int twin_index;     /* Index des kleineren Zwilling */
    int a0;             /* Zahl, fr die der ULAM_Max-Wert berechnet wird */

    /* limit muss mind. 1 sein, um Zwillinge berechnen zu knnen. */
    if (limit < 1)
    {
        return -1;
    }

    /*
     * Da das letzte Paar gesucht wird, wird ausgehend von limit, d.h. der
     * oberen Grenze des Intervalls, iteriert. Das erste gefundene Paar ist
     * dann das letzte im Intervall. Aus Effizienzgrnden wird in der gesamten
     * Iteration der ULAM-Wert fr jedes a0 nur einmal berechnet.
     */
    twin_index = -1;
    ulam_max_1 = ulam_max(limit);

    for (a0 = limit - 1; a0 >= 0 && twin_index == -1; a0--)
    {
        ulam_max_2 = ulam_max(a0);
        if (ulam_max_1 == ulam_max_2)
        {
            twin_index = a0;
        }
        else
        {
            ulam_max_1 = ulam_max_2;
        }
    }

    return twin_index;
}

/* ------------------------------------------------------------------------
 * Funktion: ulam_multiples
 * ------------------------------------------------------------------------ */
int ulam_multiples(int limit, int number)
{
    int ulam_max_1;      /* ULAM-Max-Werte von zwei benachbarten Zahlen */
    int ulam_max_2;
    int count;           /* Anzahl der bereits gefundenen Mehrlinge */
    int multiples_index; /* Index des kleinsten Mehrlings */
    int a0;              /* Zahl, fr die der ULAM-Maxwert berechnet wird */

    /* Es mssen mind. Zwillinge gesucht werden und das Intervall muss mind. so
     * viele Werte enthalten wie Mehrlinge gesucht werden */
    if (number < 2 || limit < number)
    {
```

```
        return -1;
    }

    /*
     * Da die letzten Mehrlinge gesucht werden, wird ausgehend von limit, d.h.
     * der oberen Grenze des Intervalls, iteriert. Die ersten gefundenen
     * Mehrlinge sind dann die letzten im Intervall.  Aus Effizienzgrnden wird
     * in der gesamten Iteration der ULAM-Wert fr jedes a0 nur einmal berechnet.
     */
    multiples_index = -1; /* Wert -1 bedeutet: noch kein Mehrling gefunden */
    ulam_max_1 = ulam_max(limit);
    count = 1;

    for (a0 = limit - 1; a0 >= 0 && multiples_index == -1; a0--)
    {
        ulam_max_2 = ulam_max(a0);
        if (ulam_max_1 == ulam_max_2)
        {
            /* einen weiteren gleichen ULAM-Wert gefunden */
            count += 1;
        }
        else
        {
            /* Aufeinanderfolgende Werte sind ungleich, daher Suche ab
             * aktuellem Index neu beginnen */
            ulam_max_1 = ulam_max_2;
            count = 1;
        }

        if (count == number)
        {
            /* Mehrling gefunden */
            multiples_index = a0;
        }
    }

    return multiples_index;
}


/**
 * Testet die Ulam-Funktionen mit den Beispielen vom bungszettel
 *
 * @return IMMER 0
 */
int main(void)
{
    assert(ulam_max(5) == 16);
    assert(ulam_max(7) == 52);

    assert(ulam_twins(6) == 5);
    assert(ulam_twins(5) == -1);

    assert(ulam_multiples(  10, 2) ==   5);
    assert(ulam_multiples(1000, 3) == 972);
    assert(ulam_multiples( 108, 3) ==  -1);
    assert(ulam_multiples( 391, 6) == 386);

    return (EXIT_SUCCESS);
}
```

```
+------------------------------------------------------------+
| Benoetigte Zeit (gesamt)                                   |
+------------------------------------------------------------+


real    0m9.437s
user    0m0.672s
sys     0m7.248s
```