Universität Augsburg
Institut für Informatik

Deckblatt: Übung zur Vorlesung Informatik 1 WS 2017/18
Fakultät für Angewandte Informatik
Lehrprofessur für Informatik
Prof. Dr. Lorenz, Marius Brendle, Johannes Metzger, Lev Sorokin

---

Hinweis: Es sind alle Felder auszufüllen! Abgabe der Übungsblätter immer **mittwochs** (Ausnahme wenn Feiertag: donnerstags) bis **spätestens 12:00 Uhr** in die entsprechend gekennzeichneten Briefkästen der Veranstaltung im Erdgeschoss des Instituts für Informatik (Gebäude N). Zuwiderhandlung wird mit Strafe geahndet! (Punktabzug)

Übungsblatt	

(hier die Nummer des bearbeiteten **Übungsblatts eintragen**)

	Übung 01 (1055 N) Montag 08:15 - 09:45 Uhr (Lennart Eing)
	Übung 02 (1056 N) Montag 08:15 - 09:45 Uhr (Alexander Fuchs)
	Übung 03 (1057 N) Montag 08:15 - 09:45 Uhr (Michelle Lienhart)
	Übung 04 (1055 N) Montag 12:15 - 13:45 Uhr (Henning Cui)
	Übung 05 (1056 N) Montag 12:15 - 13:45 Uhr (Christian Schavitz)
	Übung 06 (1055 N) Montag 14:00 - 15:30 Uhr (Maximilian Demmler)
	Übung 08 (1056 N) Montag 17:30 - 19:00 Uhr (Moritz Feldmann)
	Übung 09 (1057 N) Montag 17:30 - 19:00 Uhr (Dat Le Thanh)
	Übung 10 (1057 N) Dienstag 12:15 - 13:45 Uhr (Alexander Szöke)
	Übung 11 (1057 N) Dienstag 14:00 - 15:30 Uhr (Denise Böhm)
	Übung 12 (1056 N) Dienstag 17:30 - 19:00 Uhr (Marvin Drexelius)
	Übung 13 (1057 N) Dienstag 17:30 - 19:00 Uhr (Tom Wolfskämpf)
	Übung 14 (1055 N) Mittwoch 08:15 - 09:45 Uhr (Jonas Junge)
	Übung 15 (1055 N) Mittwoch 10:00 - 11:30 Uhr (Elisabeth Korndörfer)
X	Übung 16 (1054 N) Donnerstag 14:00 - 15:30 Uhr (Florian Magg)
	Übung 17 (1057 N) Donnerstag 14:00 - 15:30 Uhr (Lukas Lodes)
	Übung 18 (1054 N)* Donnerstag 17:30 - 19:00 Uhr (Patrick Eckert)
	Übung 19 (1058 N) Freitag 08:15 - 09:45 Uhr (Lena Tikovsky)
	Übung 20 (1054 N) Freitag 10:00 - 11:30 Uhr (Felix Fischer)
	Übung 21 (1055 N)* Freitag 14:00 - 15:30 Uhr (Isabell Rücker)
	Übung 23 (1057 N) Freitag 15:45 - 17:15 Uhr (André Schweiger)

(hier die eingeteilte **Übungsgruppe ankreuzen**) *(1056 N bis 03.11.17)

Teamnummer	5

(hier die Nummer des eingeteilten **Teams eintragen**)

Benjamin Ritter
Marina Huber
Anton Lydike

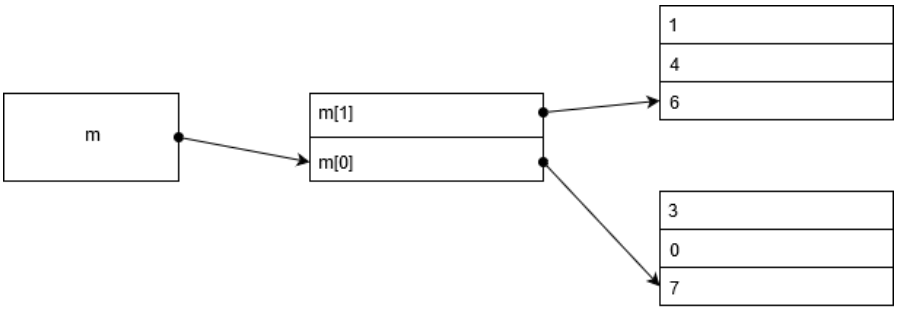
(hier die **Vor- und Nachnamen** aller Teammitglieder **eintragen**)

Aufgabe		
Aufgabe		
Aufgabe		
Aufgabe		
Gesamt		(vom Tutor auszufüllen)

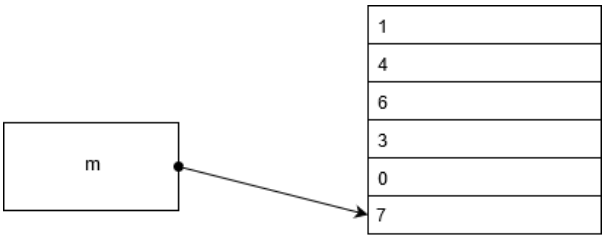
Assignment 10

37)

a)



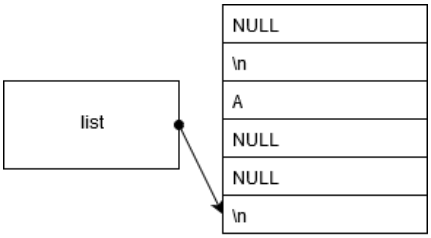
b)



c)

7
1
NULL
NULL
0
4
NULL
NULL
3
6
NULL
NULL

d)



e)

```
int main()
{
        int **A, **B;

        A = matrix_create(5, 5);

        if (!A)
                return 1;

        A = matrix_init(A, 5, 5);
        B = matrix_create(5, 5);

        if (!A) {
                matrix_destroy(A, 5);
                return 1;
        }

        matrix_init(B, 5, 5);
        matrix_print(A, 5, 5);
        matrix_print(B, 5, 5);
        matrix_destroy(A, 5);
        matrix_destroy(B, 5);

        return 0;
}
```

38)

a)

```
int split(char * w, char c, char **rest)
{
        int i = 0;

        if (rest == NULL) {
                return 0;
        }

        while (w[i] != c && w[i] != '\0') {
                ++i;
        }

        if (w[i] == c) {
                w[i] = '\0';
                *rest = &w[i + 1];
                return 1;
        } else {
                return 0;
        }
}
```

b)

```
int matrix_create(int ***m, int ze, int sp)
{
        int i, k;

        *m = malloc(ze * sizeof(int*));
        if (!(*m))
                return 0;

        for (i = 0; i < ze; i++) {
                (*m)[i] = malloc(sp * sizeof(int));
                if (!(*m)[i]) {
                        for (k = 0; k < i; ++k)
                                free((*m)[k]);

                        free(*m);
                        return 0;
                }
        }

        return 1;
}
```

c)

```
int main (void) {
        long outlong = 0;
        char * str = malloc((MAX_INPUT_LENGTH + 1) * sizeof(char));
        char * outstr = NULL;

        char buff;
        int i = 0;

        while (i < MAX_INPUT_LENGTH && (buff = getchar()) != '\n') {
                if (buff == EOF) {
                        printf("EOF encountered.\n");
                        free(str);
                        return 1;
                }

                str[i++] = buff;

        }
        str[i] = '\0';

        outlong = strtol(str, &outstr, 16);

        printf("long: %ld\nstr:  %s\n", outlong, outstr);

        free(str);

        return 0;
}
```

39)

a)

```
int **matrix_cpy_deep(int **m, int ze, int sp)
{
        int **n, i, k;

        n = malloc(ze * sizeof(int*));
        if (!n)
                return 0;

        for (i = 0; i < ze; i++) {
                n[i] = malloc(sp * sizeof(int));
                if (!n[i]) {
                        for (k = 0; k < i; k++)
                                free(n[k]);

                        free(n);
                        return NULL;
                }
                for (k = 0; k < sp; k++) {
                        n[i][k] = m[i][k];
                }
        }

        return n;
}
```

b)

```
void matrix_add(int a[][MAX_COL], int b[][MAX_COL], int n, int m)
{
        int i, j;

        for (i = 0; i < n; i++) {
                for (j = 0; j < m; j++) {
                        a[i][j] += b[i][j];
                }
        }
}
```

c)

```
void stringlist_swap(char *list[], int i, int j)
{
        char * tmp = list[i];
        list[i] = list[j];
        list[j] = tmp;
}
```

d)

```
void matrix_init_mult(int *m, int n)
{
        int i, j;

        for (i = 0; i < n; i++) {
                for (j = 0; j < n; j++) {
                        n[i*n + j] = i * j;
                }
        }
}
```

40)

c)

```
int funktion(int folge[], int n, int k)
{
        int i = 1;
        int c = 0;

        while (i <= n){
                if (folge[i-1] <= k){
                        c++;
                }
                i++;
        }

        return c;
}
```