

## **ALGORITHMEN UND DATENSTRUKTUREN**

ÜBUNG 4: PROGRAMMIEREN MIT C

Eric Kunze

eric.kunze@mailbox.tu-dresden.de

TU Dresden, 20.11.2020

C

- ► Input / Output: #include <stdio.h>
- ► Variablentypen: z.B. int, float, char
- ▶ arithmetische Operatoren: +, -, \*, /, %
- ► Vergleichsoperatoren: ==, <, <=, >, >=
- Logikoperatoren:
  - ▷ Negation: !

  - ▷ Disjunktion: ||
- ► Arrays: int feld[7] (Indizierung beginnend bei 0)

#### **INPUT & OUTPUT**

- ▶ Einlesen: scanf("%d", &n);
- ► Ausgeben: printf("n!=%d\n", factorial);

#### weiter Informationen:

```
https:
```

//www.tutorials.at/c/03-dateneingabe-ausgabe.html

#### BEDINGUNGEN

► if-else-Statement: bedingte Ausführung eines Statements

```
1 if ( BoolExp ) {
2   Statement ;
3 } else {
4   Statement ;
5 }
```

► switch-Statement: Fallunterscheidung mit mehr als zwei Fällen

```
switch ( Exp ) {
case 0: StatementSeq ; break ;
case 1: StatementSeq ; break ;
default: StatementSeq ;
}
```

#### **SCHLEIFEN**

► while-Statement: wiederholte Ausführung eines Statements (Schleifenrumpf)

```
while ( BoolExp ) {
   Statement ;
}
```

do-while-Statement: vergleichbar mit While-Statement, aber Schleifenbedingung wird nach Rumpf geprüft

```
do {
   Statement;
} while ( BoolExp )
```

#### **SCHLEIFEN**

► | for-Statement: | vor der Schleife steht Anzahl der Schleifendurchläufe fest

```
for ( Assignment ; BoolExp ; Assignment ) {
   Statement ;
}

zum Beispiel:

for (i = 1; i <= n; i = i + 1) {
   printf("i==_%d\n", i);
}</pre>
```

# Übungsblatt 4

## **AUFGABE 1 — TEIL (A)**

```
Maximum: max(n, m) = \begin{cases} m & \text{wenn } n < m \\ n & \text{sonst} \end{cases}
```

```
#include <stdio.h>
 int main() {
   int x, y, m;
  scanf("%d", &x);
   scanf("%d", &y);
   if (x < y)
   m = y;
   \} else \{ // x>=y
8
    m = x;
    printf("Das_Maximum_von_%d_und_%d_ist_%d.\n", x, y,
       m);
    return 0;
13
```

## **AUFGABE 1 — TEIL (B)**

## **Fakultät:** $n! = n \cdot (n - 1) \cdot (n - 2) \cdot \dots \cdot 2 \cdot 1$

```
#include <stdio.h>
 int main(){
   int n, i, factorial = 1;
   scanf("%d", &n);
   for (i = 1; i \le n; i = i + 1)
     factorial = factorial * i;
   printf("n!_=_%d\n", factorial);
8
   return 0;
10
```

## **AUFGABE 1 — TEIL (C)**

```
#include <stdio.h>
 int main(){
   int i,j,n;
   scanf("%d", &n);
   for ( i=1; i<=n ; i=i+1 ) {
     for (j=1; j<=n; j=j+1)
       printf("%4d", i*j);
8
     printf("\n");
   return 0;
12 }
```

## **AUFGABE 1 — TEIL (D)**

```
1 #include <stdio.h>
2 int main(){
    for (int i = 2; i <= 1000; i++) {
      int j = 2, prime = 1;
      while (j*j \ll i)
        if (i%j == 0) {
          prime = 0;
8
          break; // nicht notwendig
      j++;
      if (prime == 1)
      printf("%d_ist_prim\n", i);
    return 0;
16 }
```

#### **AUFGABE 2**

```
► A: matches > 0
▶ B: turn == 1
► C:
  z = (matches - 1) % 4;
  if (z == 0)
   z = 1
  matches = matches - z;
  turn = 0;
▶ D:
  printf("Der_Computer_zog_%d_Strichhoelzer;\n", z);
  printf("somit_werbleiben_%d_Streichhoelzer.\n", matches):
F:
  scanf("%d", &z);
  matches = matches - z;
  turn = 1:
F:
  if (turn == 1)
    printf("Der_Computer_gewinnt.");
   else
    printf("Der_Mensch_gewinnt.");
```