

Matrikelnr: _____ Name: _____ Punkte: _____

Note: _____

Handschriftlicher Teil

Die Bewertung von Lücken erfolgt ausschließlich auf Basis der *korrekt* ausgefüllten Lücken. Falsch ausgefüllte Lücken werden ignoriert und führen nicht zu Punktabzügen, d.h. **es macht keinen Unterschied, ob sie eine Lücke falsch ausfüllen oder unausgefüllt lassen.**

Karels Bedingungen

onBeeper()
beeperAhead()

leftIsClear()
frontIsClear()
rightIsClear()

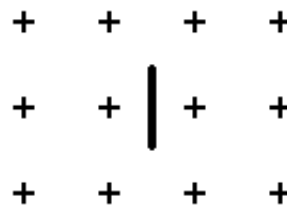
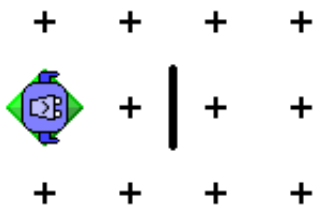
Aufgabe 1 (4 Punkte)

Gegeben sei folgendes Karel-Programm:

```
void twix()
{
    pickBeeper();
    turnLeft();
    raider();
    dropBeeper();
    raider();
}
```

```
void raider()
{
    moveForward();
    turnRight();
    moveForward();
    moveForward();
    moveForward();
    turnRight();
    moveForward();
}
```

Zeichnen Sie den Zustand der Welt (d.h. Karels Position und Blickrichtung sowie die in der Welt herumliegenden Diamanten) nach Ausführung von `twix` in das Diagramm ein:



Zustand der Welt vor Programmausführung

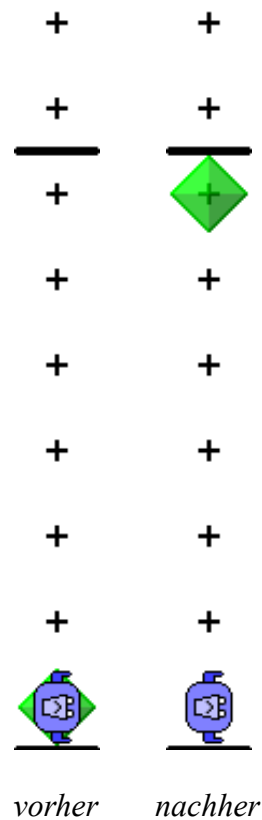
Zustand der Welt nach Programmausführung

Aufgabe 2 (7 Punkte)

Vervollständigen Sie das folgende Programm, mit dem Karel einen Lampion an der Decke befestigt und anschließend zum **Boden** zurückkehrt. Die Decke kann dabei **beliebig hoch** sein:

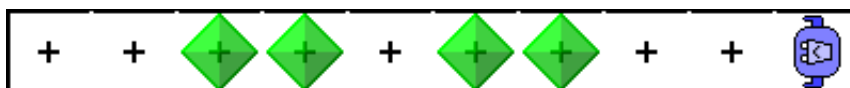
```
void moveToWall()
{
    while (frontIsClear())
    {
        moveForward();
    }
}

void hangOneLampion()
{
    _____
    _____
    _____
    _____
    _____
    _____
    _____
}
```

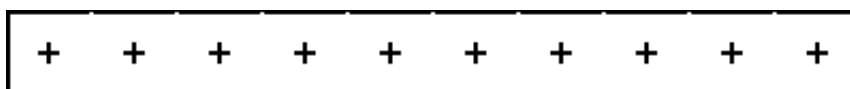


Aufgabe 3 (6 Punkte)

Zeichnen Sie den Zustand der Welt nach Ausführung von `decrement` ein. (Dieser Befehl subtrahiert 1 von einer binär dargestellten Zahl x , wobei $x > 0$.) Vervollständigen Sie anschließend den Rumpf:



vorher

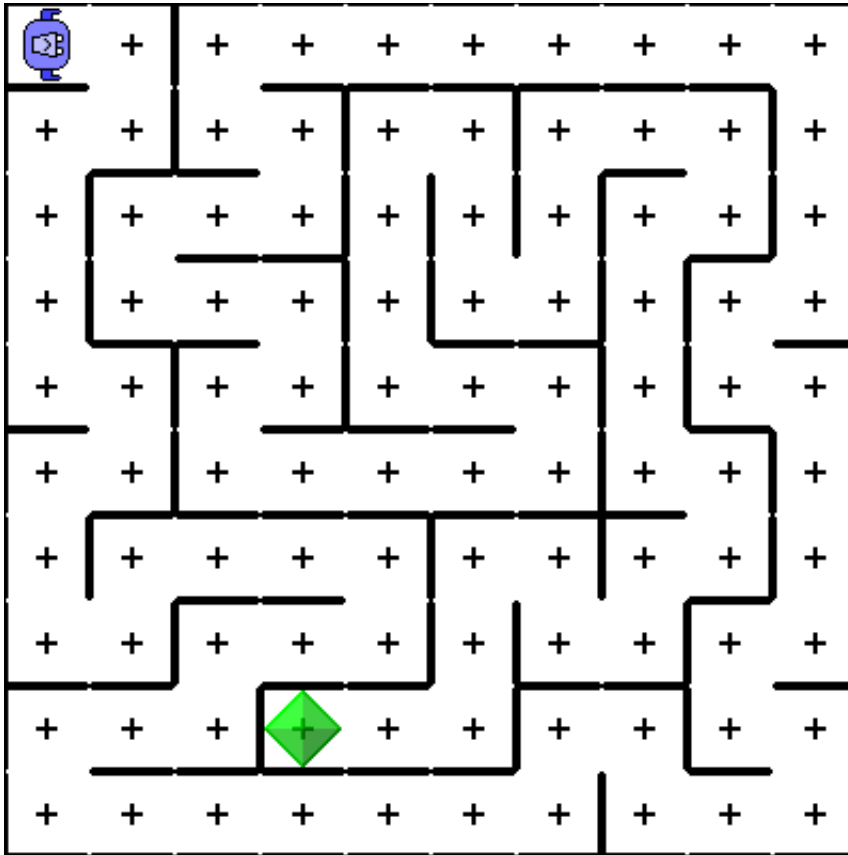


nachher

```
void decrement()
{
    while (_____)
    {
        _____
        _____
    }
    _____
}
```

Aufgabe 4 (6 Punkte)

Helfen Sie Karel, den Diamanten in Labyrinthen **ohne Abzweigungen** zu finden. Es gibt nur zwei Sackgassen; in der ersten startet Karel, und in der zweiten befindet sich der gesuchte Diamant.



```
void walkTheLabyrinth()
{
    repeat (_____)
    {
        turnAwayFromWall();

        _____
    }
}
```

```
void turnAwayFromWall()
{
    if (_____)
    {
        if (_____)
        {
            _____;
        }
        else
        {
            _____;
        }
    }
}
```

Aufgabe 5 (7 Punkte)

Studierenden Sie das folgende Programm:

```
#include <stdio.h>
#include <stdlib.h>
#pragma warning(disable: 4996)

int main()
{
    int n;
    double m;

    printf("Wie viele Kinder haben Sie? ");
    scanf("%d", &n);

    printf("Wie viel Geld haben Sie pro Monat übrig? ");
    scanf("%lf", &m);

    printf("Jedes der %d Kinder bekommt %.2f Taschengeld.\n", n, m / n);

    system("pause");
    return 0;
}
```

Wie viele Funktionen werden in diesem Programm definiert? _____

Wie viele Funktionsaufrufe befinden sich innerhalb von main? _____

Wie viele Variablen werden in diesem Programm definiert? _____

Wie viele Argumente werden bei der Frage nach dem Geld an scanf übergeben? _____

Worin unterscheiden sich die beiden Datentypen int und double? (1 Satz)

Welche Bedeutung hat das \n bei printf? (1 Satz)

Welche Rolle spielt das Prozentzeichen % bei printf? (2 Sätze)

Aufgabe 6 (4 Punkte)

Beantworten Sie die 4 Fragen im unten stehenden Quelltext:

```
int i;
for (i = 7; i <= 19; ++i)
{
    // Wie oft wird der Schleifenrumpf betreten? ____
}

for (i = 10; i == 10; ++i)
{
    // Wie oft wird der Schleifenrumpf betreten? ____
}

// Wie oft wird die äußere Schleifenbedingung geprüft? ____
for (i = 0; i < 9; ++i)
{
    int k;
    // Wie oft wird die innere Schleifenbedingung INSGESAMT geprüft? ____
    for (k = 0; k < 9; ++k)
    {
    }
}
```

Aufgabe 7 (2 Punkte)

Was unterscheidet die do-while-Schleife von der while-Schleife? (1-2 Sätze)

Aufgabe 8 (4 Punkte)

Studieren Sie die folgende Funktion:

```
int f()
{
    return rand() % 100;
}
```

Was ist die kleinste Zahl, die `f()` liefern kann? _____

Was ist die größte Zahl, die `f()` liefern kann? _____

Angenommen, `rand()` liefert Zufallszahlen bis 32767, und alle Zahlen kommen mit derselben Wahrscheinlichkeit vor. Sind die Zahlen, die `f()` liefert, dann ebenfalls gleich verteilt? (1-2 Sätze)

Aufgabe 9 (6 Punkte)

Es haben sich 3 Fehler in die Funktion `reverse` eingeschlichen, welche eine Zeichenkette umdrehen soll. Markieren Sie die fehlerhaften Stellen und beschreiben Sie jeweils das Problem.

```
#include <string.h>

void reverse(char s[])
{
    int n = strlen(s);
    int i, k;
    for (i = 0, k = n; i < n; ++i, --k)
    {
        s[i] = s[k];
        s[k] = s[i];
    }
}
```

Aufgabe 10 (4 Punkte)

Wie viele Punkte bekommt der Spieler laut `update_score` für einen Tetris (4 Zeilen)? _____

```
void update_score(int new_lines)
{
    switch (new_lines)
    {
        case 4: score += 400;
        case 3: score += 300;
        case 2: score += 200;
        case 1: score += 100;
    }
}
```

Mit einem Array lässt sich dieselbe Funktion deutlich eleganter lösen. Vervollständigen Sie:

```
void update_score(int new_lines)
{
    int line_score[] = { _____ };
    score += line_score[new_lines];
}
```

Visual Studio Teil

Bitte schreiben Sie **alle geforderten Funktion in dieselbe C-Datei** hinein! Am Ende soll auch nur diese eine C-Datei hochgeladen werden. Legen Sie also bitte auch nur **ein einziges Projekt** an.

Zum Testen der geforderten Funktionen während der Klausur empfiehlt es sich, eine `main`-Funktion zu schreiben. Ob sie diese `main`-Funktion am Ende der Klausur in der Lösung belassen oder nicht, spielt keine Rolle. Die Bewertung erfolgt ausschließlich aufgrund der geforderten Funktionen.

Keine der geforderten Funktionen soll auf die Konsole schreiben oder von der Konsole lesen. Wenn Sie `printf` oder `scanf` in diesen Funktionen verwenden, dann sind Sie auf dem Holzweg!

Wie erstelle ich ein neues Projekt in einer neuen Projektmappe?


Datei / Neu / Projekt...

Installierte Vorlagen: Visual C++ / Allgemein / Leeres Projekt

Name: Klausur

Projektmappenname: Klausur


OK

Projektmappenexplorer: Projektmappe "Klausur" / Klausur / Quelldateien  / Hinzufügen / Neues Element...

Installierte Vorlagen: Visual C++ / Code / C++-Datei (.cpp)

Name: *mustermann.c* (← **Bitte hier Ihren Nachnamen mit Endung .c eintragen!**)

OK

Projektmappenexplorer: Projektmappe "Klausur" / Klausur  / Eigenschaften

Konfigurationseigenschaften / C/C++ / Allgemein / Warnstufe: Level 4 (**W4**)

Warnungen als Fehler behandeln: Ja (**WX**)

Aufgabe A (10 Punkte)

Schreiben Sie eine Funktion `void rotate_clockwise(int matrix[9])`, welche eine 3x3 Matrix um 90 Grad im Uhrzeigersinn dreht. Anwendungsbeispiel:

```
int m[] = { 90, 56, 27, 89, 68, 98, 17, 52, 22 };
rotate_clockwise(m);
printf("%d %d %d %d %d %d %d %d %d\n",
       m[0], m[1], m[2], m[3], m[4], m[5], m[6], m[7], m[8]);
```

Das `printf` im obigen Beispiel sollte folgendes auf die Konsole schreiben:

```
17 89 90 52 68 56 22 98 27
```

Aufgabe B (10 Punkte)

Schreiben Sie eine Funktion `int is_non_descending(int a[], int n)`. Diese Funktion soll überprüfen, ob das übergebene Array `a` der Länge `n` in nicht-absteigender Reihenfolge geordnet ist, d.h. ob jede Zahl in dem Array mindestens so groß ist wie ihr linker Nachbar. Falls das Array entsprechend geordnet ist, soll die Funktion 1 liefern, ansonsten 0.

Aufgabe C (10 Punkte)

Header-Dateien der C-Standardbibliothek haben in C++ eine abweichende Namenskonvention:

```
"stdio.h"    // C   Norm: mit .h am Ende
"cstdio"     // C++ Norm: mit c am Anfang
```

Schreiben Sie eine Funktion `void convert_header_name(char s[])`, die Header-Datei-Namen der C-Norm in die entsprechende C++ Norm konvertiert. Anwendungsbeispiel:

```
char a[] = "stdio.h";
convert_header_name(a);
printf("%s\n", a);    // cstdio
```

Aufgabe D (10 Punkte)

Schreiben Sie eine Funktion `void count(char s[], int n[128])`, welche die Häufigkeit der in der Zeichenkette `s` vorkommenden Buchstaben im Array `n` ablegt. Anwendungsbeispiel:

```
int vorkommen[128];
count("erdbeere", vorkommen);
printf("%c kommt %d Mal vor\n", 'e', vorkommen['e']);    // e kommt 4 Mal vor
```

Aufgabe E (10 Punkte)

Schreiben Sie eine Funktion `int anagrams(char a[], char b[])`. Diese Funktion soll genau dann 1 liefern, wenn `a` ein Anagramm von `b` ist, d.h. wenn jeder Buchstabe genau so oft in `a` vorkommt wie in `b`. Ansonsten soll die Funktion 0 liefern. Beispiele für Anagramme:

atlas	eichel	fernsehen	reifen	schaden
salat	leiche	ehrensenf	ferien	schande

Hochladen Ihrer Lösung am Ende der Klausur

Wenn Sie mit der Bearbeitung der Aufgaben A bis E fertig sind, dann kopieren Sie die Datei `nachname.c` in den Ordner `L:\PROF\abu349\Stud\WiSe-14-15\E1b-PRP1\Ihr-Name`.

Falls Sie Ihre Lösung nicht im Dateisystem lokalisieren können, dann markieren Sie einfach den gesamten Quelltext mit Strg A, kopieren ihn mit Strg C, starten Notepad (im Windows-Startmenü *Editor* eingeben), fügen ihn mit Strg V ein und speichern ihn auf dem Desktop als `nachname.c`. Anschließend verschieben Sie diese neue Datei in das oben beschriebene Verzeichnis auf L:.
Wenn auch das nicht funktionieren mag, bitten Sie einen der anwesenden Assistenten um Hilfe.

Hamburg, den 5. Februar 2015 Unterschrift: _____