



Exercise Session — Computer Science — 13

Adel Gavranović

Rätsel, Knackpunkt(e), Fragerunde

Übersicht

Follow-up

Memory Management / Bug Hunt

Alte Prüfungen

Riddle

Prüfungsvorbereitung

Knackpunkte

if-Klauseln vereinfachen



`n.ethz.ch/~agavranovic`

 Material

 Webpage

 Mail

1. Intro

Intro

- Letzte Übungsstunde gemeinsam :(

2. Follow-up

2. Follow-up

2.1. Memory Management / Bug Hunt

Bug Hunt I

```
// PRE: len is the length of the memory block that starts at array
void test1(int* array, int len) {
    int* fourth = array + 3;
    if (len > 3) {
        std::cout << *fourth << std::endl;
    }
    for (int* p = array; p != array + len; ++p) {
        std::cout << *p << std::endl;
    }
}
```

Finde Fehler im Code und schlage Korrekturen vor

Bug Hunt I – Dangerous Pointer

```
// PRE: len is the length of the memory block that starts at array
void test1(int* array, int len) {
    //int* fourth = array + 3;    // ERROR
    if (len > 3) {
        int* fourth = array + 3;    // OK
        std::cout << *fourth << std::endl;
    }
    for (int* p = array; p != array + len; ++p) {
        std::cout << *p << std::endl;
    }
}
```

Auch wenn der Pointer nicht dereferenziert wird, muss er in einen Speicherblock oder auf das Element unmittelbar nach dessen Ende zeigen.

Bug Hunt II

```
// PRE: len >= 2
int* fib(int len) {
    int* array = new int[len];
    array[0] = 0; array[1] = 1;
    for (int* p = array+2; p < array + len; ++p) {
        *p = *(p-2) + *(p-1); }
    return array; }

void print(int* array, int len) {
    for (int* p = array+2; p < array + len; ++p) {
        std::cout << *p << " ";
    }
}

void test2(int len) {
    int* array = fib(len);
    print(array, len);
}
```

Bug Hunt II – Memory Leak

```
// PRE: len >= 2
int* fib(int len) {
    int* array = new int[len];
    array[0] = 0; array[1] = 1;
    for (int* p = array+2; p < array + len; ++p) {
        *p = *(p-2) + *(p-1); }
    return array; }

void print(int* array, int len) {
    for (int* p = array+2; p < array + len; ++p) {
        std::cout << *p << " ";
    }
}

void test2(int len) {
    int* array = fib(len);
    print(array, len);
    delete[] array;           // otherwise array is leaked!
}
```

Bug Hunt III

```
// PRE: len >= 2
int* fib(int len) {
    // ...
}
void print(int* m, int len) {
    for (int* p = m+2; p < m + len; ++p) {
        std::cout << *p << " ";
    }
    delete m;
}
void test2(int len) {
    int* array = fib(len);
    print(array, len);
    delete[] array;
}
```

Bug Hunt III – Double Free!

```
// PRE: len >= 2
int* fib(int len) {
    // ...
}

void print(int* m, int len) {
    for (int* p = m+2; p < m + len; ++p) {
        std::cout << *p << " ";
    }
    delete[] m;
}

void test2(int len) {
    int* array = fib(len);
    print(array, len);
    // delete[] array;           // array deallocated twice!
}
```

Fragen/Unklarheiten?

3. Lernziele

Ziele

- Wissen, wo man alte Prüfungen findet und mit ihnen üben kann
- Knackpunkte geklärt oder zumindest richtige Ressourcen gefunden
- Sich für das heutige *rw::gettogether* angemeldet

4. Feedback zu **code** expert

"Rekursive Lösungen"

- Viele haben in ihren Lösungen zu den "rekursiven" Aufgaben eine zweite Funktion eingeführt welche einen Teil der Rekursion für Spezialfälle übernimmt. Das ist suboptimal und nicht "richtig" rekursiv
- Studiert nach Abgabe der Rekursionsaufgaben unbedingt die Musterlösungen. Sie sind oft sehr elegant implementiert und erlauben es, sie auf ähnliche Aufgabenstellungen zu übertragen
- Das Muster, das wir gelernt und geübt haben (Base Case und dann Recursive Case mit Leap of Faith) ist praktisch immer¹ anwendbar

¹keine Garantien meinerseits

5. Alte Prüfungen

Mit alten Prüfungen üben

- Auf der Website des Kurses sind zahlreiche alte Prüfungen zu finden
- Sie können eine gute Möglichkeit sein, für die Prüfung zu üben
- Stellt sicher, dass ihr die richtigen anschaut (die für euren Kurs)



`lec.inf.ethz.ch/past_exams`

6. Riddle

Riddle

Riddle me dieses Codebeispiel auf **code expert** ...

7. Prüfungsvorbereitung

Prüfungsvorbereitung

■ Wissenslücken Stopfen

- findet schnellstmöglich raus, wo Ihr Wissenslücken/Skill-issues habt und geht sie an (ggf. zusammen mit anderen)

■ Alte Prüfungen Lösen

- in Prüfungssetting (@MacUsers: gewöhnt euch ans Keyboard!)
- letzten zwei spart man sich für die letzte Woche vor der Prüfung auf

■ Zusammenfassung verfassen, mit...

- Auswertungsreihenfolgen,
- komplizierte Syntax,
- Programmieraufgaben, mit denen ihr sehr viel Mühe hattet

■ weiteres?

8. Knackpunkte

Fragerunde

Woran hakt es bei dir?

(Absichtlich leer gelassen)

(Absichtlich leer gelassen)

(Absichtlich leer gelassen)

(Absichtlich leer gelassen)

(Absichtlich leer gelassen)

(Absichtlich leer gelassen)

9. if-Klauseln vereinfachen

Simpleere ifs I

Wir haben festgestellt, dass viele solchen Code schreiben

```
if (grid != nullptr) {
    if (grid->is_filled(row, col)) {
        if (col == 8) {
            if (fillValidNumber(grid, row + 1, 0)) {
                return true;
            }
        } else {
            if(fillValidNumber(grid, row, col + 1)) {
                return true;
            }
        }
        return false;
    }
}
```

Simplere ifs II

Da die Funktion `fillValidNumber` einen booleschen Wert liefert und wir auch einen Booleschen Wert zurückgeben wollen, können wir die verschachtelte `if`-Klausel und das letzte `return` entfernen

```
if (grid != nullptr) {  
    if (grid->is_filled(row, col)) {  
        if (col == 8) {  
            return fillValidNumber(grid, row + 1, 0);  
        } else {  
            return fillValidNumber(grid, row, col + 1);  
        }  
    }  
}
```

Simplere ifs III

Wir können auch die Tatsache nutzen, dass `&&` einen Kurzschluss darstellt, um die die beiden äusseren `if`-Anweisungen zusammenzuführen

```
if (grid != nullptr && grid->is_filled(row, col)) {  
    if (col == 8) {  
        return fillValidNumber(grid, row + 1, 0);  
    } else {  
        return fillValidNumber(grid, row, col + 1);  
    }  
}
```

Fragen/Unklarheiten?

10. Outro

Allgemeine Fragen?

Bis zum nächsten Mal

Bis heute Abend und viel Erfolg bei den Prüfungen!