Programmierkurs IPK WS22/23
Dr. Dominic Kempf
IWR, Universität Heidelberg

Probeklausur
Abgabedatum Keine Abgabe

Probeklausur

Allgemeine Hinweise:

- Dieses Blatt wird nicht abgegeben. Es dient lediglich zu Ihrer Orientierung was Sie in der Klausur erwartet. Fragen können Sie sowohl in den Tutorien als auch in der Vorlesungssitzung vom 10. Februar stellen.
- Die Klausur wird insgesamt sechs Aufgaben enthalten, welche alle gleich viele Punkte geben.
- Die Aufgaben 1-3 auf diesem Blatt werden genau so in der Klausur mit anderen Teilaufgaben (variierendem Schwierigkeitsgrads) gestellt werden.
- Darüber hinaus werden drei weitere Aufgaben gestellt. Aufgabe 4 dieser Probeklausur ist ein Beispiel für eine Programmieraufgabe, es werden aber auch die Nicht-Programmierthemen der Vorlesung behandelt.
- Zum Bestehen der Klausur sind 50% der Punkte hinreichend.

Aufgabe 1: Verständnisfragen

Geben Sie **kurze** Antworten auf die folgenden Fragen:

- Welche zwei Outputstreams sind mit jedem Unix Programm verbunden?
- Was ist die Aufgabe eines Destruktors und wie heißt er für eine Klasse `MyClass`

In der Klausur werden 10 Teilaufgaben dieser Art gestellt.

Aufgabe 2

In den folgenden Codebeispielen dürfen Sie davon ausgehen, dass die nötigen Header inkludiert wurden. Sie dürfen zudem davon ausgehen, dass die gegebenen Funktionen mit passenden Argumenten aufgerufen werden, auch wenn die `main` nicht abgedruckt ist.

Die Programme enthalten alle exakt einen Fehler. Beschreiben Sie die Fehler in wenigen Worten und beantworten Sie die Frage, ob es sich dabei um Kompilierfehler oder Laufzeitfehler handelt. Beachten Sie dabei die laut Kommentaren erforderliche Funktionalität.

Hinweis: Sie müssen die Fehler nicht korrigieren, es geht darum sie zu beschreiben. Beispiel: “In Zeile 15 wird die Variable `foo` benutzt, welche nie deklariert wurde. Kompilierfehler.”

- Programmcode:

```
int i = 0;
for (int j=0; i<100; j++)
  i *= j;
std::cout << i << std::endl;
```

- Programmcode:

```
class Object {
  public:
  Object(int x) {
    std::cout << "Constructed object with " << x << std::endl;
  }
```

```
};

int main() {
  Object o;
}
```

In der Klausur werden 5 Teilaufgaben dieser Art gestellt.

Aufgabe 3

Die folgenden Codeschnipsel enthalten jeweils ein wohlgeformtes Programm. Die notwendigen Include Statements wurden für bessere Übersicht weggelassen. Geben Sie jeweils die Ausgabe des Programms an. Sie können Ihre Lösung neben das Programm schreiben.

1. Programmcode:

```
int main() {
  int a = 3;
  for (int i = 0; i<3; ++i)
  {
    int a = i;
    a += i;
  }
  std::cout << a << std::endl;
}
```

2. Programmcode:

```
int main() {
  std::vector<int> data{1,2,3,4,5};
  std::vector<int> transformed;
  auto scale = [](int i){ return 2 * i; };
  std::transform(data.begin(), data.end(), std::back_inserter(transformed), scale);
  for (auto item: transformed)
    std::cout << item << " ";
}
```

In der Klausur werden 5 Teilaufgaben dieser Art gestellt.

Aufgabe 4

Die C++-Standardbibliothek bietet folgende Funktionen, um die Anzahl der Elemente in einer Iterator Range zu bestimmen, für welche ein gegebener Funktor `true` zurückgibt.

```
template< class InputIt, class UnaryPredicate >
typename iterator_traits<InputIt>::difference_type
count_if( InputIt first, InputIt last, UnaryPredicate p );
```

Gegeben sei nun folgende Klasse (nicht vollständig):

```
class Person {
public:
  int age() const;
};
```

Schreiben Sie eine Funktion welche für einen Container vom Typ `std::vector<Person>` die Anzahl der Personen unter 18 Jahren und die Anzahl der Personen über 65 Jahren ermittelt. Verwenden Sie dafür die folgende Signatur:

```
std::array<std::size_t, 2> young_and_old_people(const std::vector<Person>& people);
```