1. Prüfung zu VO Programmiersprache 1 17.03.2022 WS 2021/22

Angabe Nr. 2

Es sind keine Unterlagen erlaubt! Alle Multiple-Choice Aufgaben haben 4 Unterpunkte a) bis d), die entweder wahr oder falsch sind. Sie haken das Antwortkästchen deutlich an, wenn die Aussage wahr ist bzw. lassen das Kästchen frei, wenn die Aussage falsch ist.

1) Folgende Streameingabe für den Objekttyp `Complex` sei definiert:

```
class Complex {
    double re, im;
};

std::istream& operator>>(std::istream& is, Complex& c)
{   is >> c.re >> c.im;
    return is;
}
```

a) Die Streameingabe erzeugt einen Compilerfehler, weil `c.re` privat ist.

b) Die Streameingabe erzeugt einen Compilerfehler, weil `c.re` konstant ist.

c) Damit eine Streameingabe korrekt funktioniert, muss man das einzulesende Objekt als konstante Referenz übergeben werden.

d) Eine Streameingabe, bei der das einzulesende Objekt per Wert übergeben wird, kann nicht korrekt funktionieren.

2) In einem C++ Programm sei folgendes Code-Fragment enthalten:

```
int i = 7, j = 1;
double u = 1.5;
auto f = [=](auto& x) { x += i; };
f(j);
f(u);
```

a) Nach diesem Codefragment hat `u` den Wert `8.5`

b) Lambdas sind erst ab C++11 erlaubt.

c) Die Typbezeichnung `auto` für Argumente von Lambdas sind derzeit nicht erlaubt.

d) Das Gleichheitszeichen im Capture-Ausdruck hätte man hier auch weglassen können, ohne dass dadurch ein Fehler entsteht.

3) Betrachten Sie das folgende C++-Programm, bei dem ich zur besseren Orientierung Zeilennummern eingefügt habe (im Quellcode stehen natürlich keine Zeilennummern):

```
1:      class Klausur {
2:      public:
3:           int bewerte() const;
4:      };
5:
6:      int main()
7:      {    Klausur k;
8:           auto note = bewerte(k);
          ...
```

   a) Irgendwo im Quellcode muss die Funktion `int Klausur::bewerte() const` definiert werden, damit das Programm compiliert.
   b) Diese Klasse besitzt einen Standardkonstruktor und einen Kopier-Konstruktor.
   c) Zeile 8 sollte lauten `auto note = k.bewerte();`
   d) Keine andere Klasse, z.B. `class Pruefung` darf im selben Programm eine Methode `int bewerte() const` haben.

4) Kreuzen Sie alle korrekten Codefragmente an, die eine Python-`list x` mit den Zahlen 1, 2, 3, 4, 5 erzeugen:
   a) `x = [ 1, 2, 3, 4, 5]`
   b) `x = [ range(5) ]`
   c) `x = [ range(1, 6)]`
   d) `x = list(range(1,6))`

5) Folgende 2 C++-Funktionen sind im selben Programm definiert
   i) `int f(int x) { return x; }`
   ii) `double f(int y) { return y; }`
   a) Die beiden Definitionen sind erlaubt, da die Namen der Argumente (`x` und `y`) unterschiedlich sind.
   b) Wenn die Funktionen in verschiedenen Namespaces definiert sind, ist die Definition korrekt.
   c) Die beiden Definitionen sind korrekt, da die Rückgabetypen unterschiedlich sind.
   d) Der Aufruf `double x = f(1);` ist korrekt und ruft Variante ii) auf.

6) Betrachten Sie das folgende Python-Codefragment:

```
x = ( 1, 2, -3., True, "Ein Text")
```

a) `x[:-1]` ist das Tupel `x` ohne das letzte Element.
b) `x[0:]` ist das Tupel `x` ohne ihr Anfangselement.
c) `x[-1::-1]` ist das Tupel `x` in umgekehrter Reihenfolge
d) Eine Schleife der Form: `for i in x:` ist ein Fehler, weil man mit so einer Schleife nur über eine Liste iterieren kann und nicht über ein Tupel.

7) a) Python besitzt den Datentyp `bool` für Wahrheitswerte.
b) Der Ausdruck `x == 2` ist in Python ein Wert vom Typ `bool`.
c) Der Ausdruck `x == 2` ist in C++ ein Wert vom Typ `bool`.
d) Der Ausdruck `1 + True` ist in Python korrekt und ein `int` mit Wert 2.
Der Ausdruck `1 + True` ist in C++ möglicherweise inkorrekt.

8) Betrachten Sie das folgende Code-Fragment:

```
int i, a[10];
for (i = 0; i < 10; i++)
    a[i+1] = a[i];
```

a) Ein Zugriff auf `a[10]` ist ein Fehler.
b) Nach dieser Schleife haben alle Array-Elemente von `a` denselben Wert.
c) In obiger Schleife gibt es einen Bereichsüberlauf, d.h. es wird in einen nicht reservierten Bereich geschrieben.
d) Das erste Array-Element ist `a[0]`.

9) Welchen Wert haben die Variablen i, j, k nach folgendem Code-Fragment:

```
int i = 5, j = 4, k = 2;
i = (--j)/(k++);
```

a) Es gilt $i = 1$
b) Es gilt $i = 2$
c) Es gilt $j = 3$
d) Es gilt $k = 2$

10) Welche der folgenden Aussagen treffen auf C++ zu?
a) Eine `for`-Schleife wird immer wenigstens einmal durchlaufen.
b) Eine `while`-Schleife darf nicht mit `return` verlassen werden.
c) Ein `if -- else if -- else`-Block darf nicht mit `return` verlassen werden.
d) Ein `if -- else if -- else`-Block darf nicht mit `break` verlassen werden.

11) Welche der folgenden Aussagen treffen auf C++ zu:
   a) Es kann 2 positive `int` $x > 0$, $y > 0$ mit $x + y < 0$ geben.
   b) Es kann 2 positive `double` $x > 0$, $y > 0$ mit $x + y < 0$ geben.
   c) Es kann 2 positive `unsigned int` $x > 0$, $y > 0$ mit $x + y < 0$ geben.
   d) Es kann 2 positive `unsigned int` $x > 0$, $y > 0$ mit $x + y == 0$ geben.

12) Betrachten Sie das C++-Programm, in dem ich aus Platzgründen die Funktionen in eine Zeile geschrieben habe:

```
void mod3(int x)  { x = x % 3; }
void mod7(int &x) { x = x % 7; }

int main()
{       int i = 10, j = 11, k = 12;
        mod3(i); mod7(k);
}
```

   a) Am Ende der `main`-Funktion besitzt i den Wert 10.
   b) `mod7()` verwendet die Übergabe per Referenz und kann daher das Argument x verändert zurückgeben.
   c) Am Ende der `main`-Funktion besitzt k den Wert 5.
   d) Eine Variable, die per Pointer an eine Funktion übergeben wird, kann durch diese Funktion verändert werden.