Name: Matrikelnummer:

Hauptklausur: PRGIII MD

Aufgabe 1 (20 Punkte)
Folgende C++ Programme enthalten Fehler, die vom Compiler entdeckt werden. Erklären Sie den Fehler und machen Sie einen Korrekturvorschlag.

a)
```
#include <iostream>
int main(){
  int xs [] = {1,2,3,4,5};
  int ys [5]= xs;
  std::cout << ys[2];
}
```

b)
```
int main(){
  int xs = 42;   int& rXs = xs;
  rXs = 15;
  int& & rrXs = rXs;
  rrXs = 17;
}
```

c)
```
#include <iostream>
class Date{
  public:
    int year;int month; int day;
    Date(int year,int month,int day):
        year(year),month(month),day(day){};
};

int main(){
  Date d;
  std::cout << "starting main" << std::endl;
  d=Date(2004,1,26);
}
```

d)
```
#include <string>
#include <iostream>
using namespace std;

class Person{
 public:
   string name; string vorname;
   Person(string name,string vorname)
    :name(name),vorname(vorname){}
   string getFullName(){return vorname+" "+name;}
```

```
11 };
12
13 int main(){
14   Person* p = new Person("Zerlet","Helmut");
15   cout << p.getFullName();
16 }
```

e)

```
1  #include <string>
2  #include <iostream>
3  using namespace std;
4
5  class Person{
6   public:
7     string name; string vorname;
8     Person(string name,string vorname)
9      :name(name),vorname(vorname){}
10    string getFullName(){return vorname+" "+name;}
11 };
12
13 template <typename at>
14 at k(at x, at y){
15   return x+y;
16 }
17 int main(){
18   char* n ="Zerlett"; string v ="Helmut";
19   cout << k(2,40)<<endl;
20   cout << k(Person(n,v),Person(n,v)).name<<endl;
21 }
```

Aufgabe 2 (24 Punkte)
Rechnen Sie folgende Programme auf dem Papier und geben Sie die Ausgabe auf dem Bildschirm an. Erklären Sie kurz, wie es zu der Ausgabe kommt.

a)

```
1  #include <iostream>
2  int main(){
3    int xs [] = {1,3,5,7,9};
4    std::cout << *(xs+3) << std::endl;
5    std::cout << *xs+3   << std::endl;
6  }
```

b)

```
1  #include <string>
2  #include <iostream>
3  using namespace std;
4
```

```
5  class C1 {
6  public:
7     virtual string getDescription(){return "C1";}
8  };
9
10 class C2: public C1 {
11 public:
12   virtual string getDescription(){return "C2";}
13 };
14
15 int main(){
16   C1 c1; C2 c2; c1 = c2; C1* pC1 = &c2;
17
18   cout << c1.getDescription()<< endl;
19   cout << c2.getDescription()<< endl;
20   cout << pC1->getDescription()<< endl;
21   pC1 = new C2();
22   cout << pC1->getDescription()<< endl;
23 }
```

c)

```
1  #include <iostream>
2  using namespace std;
3
4  template <class At>
5  class Box{
6  public:
7    At content;
8    Box<At>(At x):content(x){};
9  };
10
11 int f(int& x){
12   x=x+4; return x;
13 }
14
15 template <typename At>
16 void fB(Box<At> b){
17   b.content  = f(b.content);
18 }
19
20 int main(){
21   int y = 42; int z = (f(y)); Box<int> b(17); fB(b);
22
23   cout << z << endl;
24   cout << y << endl;
25   cout << b.content << endl;
```

```
26 }
```

Aufgabe 3 (28 Punkte)
Schreiben Sie folgende generischen Funktionen für die STL:

a)
```
template <typename Iterator, typename Function
          ,typename ResultType>
void mapIntoVector(Iterator begin,Iterator end
                ,Function f,vector<ResultType>& result);
```

`mapIntoVector` soll die übergebene Funktion `f` auf jedes Element im Iteratorbereich anwenden und das entsprechende Ergebnis am Ende des Vektors `result` einfügen.

b) Schreiben Sie eine Beispielanwendung der obigen Funktion `mapIntoVector`, in der für eine Sammlung von Stringobjekten jeweils die Länge der einzelnen Strings in den Ergebnisvektor übertragen wird.

c)
```
template <typename Iterator, typename ElementType>
vector<ElementType>* filter
  (Iterator begin,Iterator end,bool (*f)(ElementType));
```

`filter` soll einen Zeiger auf einen Vektor zurückgeben, in dem alle Elemente des Iteratorbereichs enthalten sind, für die die Funktion `f` true als Ergebnis liefert.

Aufgabe 4 (28 Punkte)
Schreiben Sie eine generische Klasse für die Darstellung von Binärbäumen. Halten Sie die Implementierung ähnlich der Implementierung der einfach verketteten Listen in der Vorlesung.

a) Die Klasse soll generisch über den Elementtypen der in dem Baum gespeicherten Elemente sein. Es gibt zwei Arten von Baumknoten:

- Blätter: diese Knoten haben keine Kinder.
- Verzweigungen: diese Knoten können ein linkes und ein rechtes Kind haben.

b) Fügen Sie Ihrer Baumklasse eine Funktion `int size()` hinzu, die angibt, wie viele Baumknoten der Baum enthält.

c) Schreiben Sie einen kleines Testbeispiel für die Benutzung Ihrer Baumklasse.

Benotung

Aufgabe:	1	2	3	4		Summe		Übungen	Gesamt
Punkte:	20	24	28	28		100		10	
erreicht:									