2. Semester, 2. Prüfung

Name	

- Die gesamte Prüfung bezieht sich auf die Programmierung in C++!
- Prüfungsdauer: 90 Minuten
- Mit Kugelschreiber oder Tinte schreiben
- Lösungen können direkt auf die Aufgabenblätter geschrieben werden
- PC's sind nicht erlaubt
- Unterlagen und Bücher sind erlaubt
- Achte auf Details wie Punkte, Kommas und Semikolons

Aufgabe	Punkte	
Vektoren und Arrays		
Teilobjekte und statische Elemente		
Klassen und Strukturen		
Programmverständnis		
Weitere Fragen		
Total		

1. Vektoren und Arrays

a) Erzeuge ein Array von 20 longs mit dem Namen *Datenarray*. Du brauchst das Array ausnahmsweise nicht zu initialisieren. (2 Punkte)

```
long Datenarray[20];
```

b) Was hat die Variable „z“ im folgenden Beispiel für einen Wert? (2 Punkte)

```
int daten[5] = { 2, 4, 6, 7, 5 };
int z = daten[4];
```

```
z = 5
```

c) Definiere ein Array von 10 int-Werten und initialisiere es so, dass alle Elemente den Wert 0 haben. Verwende eine möglichst einfache Schreibweise (das heisst eine Zeile)! (2 Punkte)

```
int meinArray[10] = {0};
```

d) Definiere ein Array von 10 int-Werten und schreibe eine Schleife, die alle Werte initialisiert. Initialisiere die Werte so, dass das erste Element den Wert 0 hat und das letzte den Wert 9! (4 Punkte)

```
int meinArray[10];
for(long i = 0; i < 10; ++i)
{
    meinArray[i] = i;
}
```

2. Teilobjekte und statische Elemente

a) Es sind zwei Klassen gegeben: Die Klasse Auto und die Klasse Rad. Die Klasse Rad hat einen Konstruktor mit Parametern, aber keinen Default-Konstruktor. In der Klasse Auto ist ein Konstruktor mit Parametern definiert. Ein Parameter gibt die Länge des Autos an, der zweite den Radius für die Räder. Schreibe für diesen Konstruktor den Code. (4 Punkte)

```
class Rad
{
    public:
        Rad(long radius) : m_radius(radius)
        {
        }

    private:
        long m_radius;
};

class Auto
{
    public:
        Auto(long laenge, long radradius);
    private:
        Rad m_lvRad;
        Rad m_rvRad;
        Rad m_hlRad;
        Rad m_hrRad;
};



Auto::Auto(long laenge, long radius)
     :m_lvRad(radius),
      m_rvRad(radius),
      m_hlRad(radius),
      m_hrRad(radius)
{
}
```

b) Schreibe den Code für den Konstruktor für die folgenden Klasse Kreis. Es gilt dabei die Datenelemente richtig zu initialisieren. (3 Punkte)

```
class Kreis
{
    public:
        Kreis(double Radius);  // Konstruktor
    private:
        const double PI;
        double m_Radius;
};
```

```
Kreis::Kreis(double Radius)
      :PI(3.1415),
       m_Radius(Radius)
{
}
```

3. Klassen und Strukturen

a) Nimm an du arbeitest an einem Zeichnungsprogramm.
Definiere eine Struktur für eine Farbe mit drei Elementen für die drei Grundfarben rot, gruen und blau. Die jeweiligen Farben können eine Sättigung von 0 bis 255 haben, wähle also den passenden Datentypen. (2 Punkte)

```
struct Farbe
{
    unsigned char rot;
    unsigned char gruen;
    unsigned char blau;
};
```

b) Definiere eine Klasse Dreieck, mit den Datenelementen, die zu einem Dreieck gehören und füge als *zusätzliches* Datenelement die Farbe hinzu. Verwende hierfür die Struktur von Aufgabe a).
Definiere eine Konstruktor mit Parametern, aber keinen default-Konstruktor. Mit einer weiteren Methode soll die Farbe neu definiert werden können. Schreibe auch den Code für die Methoden unter die Klassedefinition. (6 Punkte)

```
struct Punkt
{
    int x;
    int y;
};

class Dreieck
{
    public:
        Dreieck(Punkt p1, Punkt p2, Punkt p3, Farbe farbe);
        void setzeFarbe(Farbe farbe);
    private:
        Punkte m_Ecken[3];
        Farbe  m_Farbe;
};

Dreieck::Dreieck(Punkt p1, Punkt p2, Punkte p3, Farbe farbe)
{
    m_Ecken[0] = p1;
    m_Ecken[1] = p2;
    m_Ecken[2] = p3;
    m_Farbe = farbe;
}
void Dreieck::setzeFarbe(Farbe farbe)
{
    m_Farbe = farbe;
}
```

4. Programmverständnis

a) Betrachte folgenden Code und finde heraus was er ausgibt. (2 Punkte)

```
#include <iostream>
using namespace std;

int main()
{
    char etwas[] = { 'B', 'L', 'A', 'H', 0, 'x', 'x' };
    cout << etwas;

    // 0 ist eine null !!

    return 0;
}
```

```
BLAH
```

b) Was gibt dieser Code aus? (2 Punkte)

```
#include <iostream>
using namespace std;

int main()
{
    int zahlen[10];
    for(long i = 0; i < 10; ++i)
    {
        zahlen[i] = i;
    }
    cout << zahlen[9] << zahlen[0] << " Minuten sind lang";

    return 0;
}
```

```
90 Minuten sind lang
```

c) Was gibt folgender Code aus? (4 Punkte)

```
#include <iostream>
using namespace std;

//////////////////////////

class Objekt
{
public:
    Objekt()
    {
        s_Anzahl++;
        m_Nummer = s_Anzahl;
    }

    void ping()
    {
        cout << m_Nummer << endl;
    }

private:
    static long s_Anzahl;
    long m_Nummer;
};

//////////////////////////

long Objekt::s_Anzahl = 0;

//////////////////////////

int main()
{
    Objekt objekte[5];

    for(int i = 0; i < 5; ++i)
    {
        objekte[i].ping();
    }

    return 0;
}
```

```
1
2
3
4
5
```

5. Weitere Fragen

a) Wieviele Objekte werden in diesem Beispiel erzeugt? (2 Punkte)

```
class Test
{
    public:
        Test();

    private:
        int m_Zahl;
};

int main()
{
    Test array[11];
    Test& erstes = array[0];
    Test& letztes = array[10];
    return 0;
};
```

11

b) Welches ist in diesem Beispiel der grösstmögliche Index **Z**? (2 Punkte)

```
char text[] = “Hallo“;
char einBuchstabe = text[Z];
```

5

c) Was hat die Variable z für einen Wert? (2 Punkte)

```
char text[] = “Mann“;
char z = text[4];
```

0