

```

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

// Datenstruktur deklarieren
struct FloatVec {
    float *vecPtr; // Zeiger auf ein float-Feld
    int maxmem;    // max. verfügbare Elemente
    int used;     // belegte Elemente
};

// Vektor initialisieren
void fv_init(FloatVec& v, int maxmem=10)
{
    // neues Speicherfeld erzeugen (alle automatisch 0.0)
    v.vecPtr = new float[maxmem];
    // max. verfügbare Elemente definieren
    v.maxmem = maxmem;
    // benutzte Elemente auf 0 setzen
    v.used = 0;
}

// den Vektorspeicher auf neue Größe vergrößern
void fv_resize(FloatVec& v, int size)
{
    // nur vergrößern, wenn notwendig
    if( size <= v.maxmem)
        return;
    // neuen Speicher erzeugen
    float *temp = new float[size];
    // altes Feld in neuen Speicher laden
    for(int i=0; i<v.used; i++)
        *(temp+i) = v.vecPtr[i];
    // alten Speicher freigeben
    delete[] v.vecPtr;
    // vecPtr auf neuen Speicher zeigen lassen
    // temp verschwindet nach verlassen der Funktion
    v.vecPtr = temp;
    // Anzahl max. verfügbarer Elemente erhöhen
    v.maxmem = size;
}

// einen Wert an den Vektor fügen
void fv_append(FloatVec& v, float wert)
{
    // prüfen, ob Speicher vergrößert werden muss
    if( v.used == v.maxmem)
        fv_resize(v, v.maxmem+1);
    // Anzahl benutzter Elemente erhöhen
    v.used++;
    // neues Element einfügen
    v.vecPtr[v.used-1] = wert;
}

```

```

// Vektor ausgeben mit gegebener Genauigkeit
void fv_print(const FloatVec& v, int precision=2)
{
    // Anzahl Nachkommastellen festlegen
    cout.precision(precision);
    for(int i=0; i<v.used; i++)
    {
        // max. 3 Zahlen pro Zeile
        if(i%3 == 0)
            cout << endl;
        // Darstellung im 15 Zeichen breiten Spalten
        // und in Gleitpunktdarstellung
        cout << setw(15) << fixed << v.vecPtr[i];
        // Adressen der Elemente im Speicher ausgeben
        cout << "(" << &v.vecPtr[i] << ")";
    }
    cout << endl;
}

// Vektor leeren
void fv_delete(FloatVec& v)
{
    // Speicherfeld freigeben
    delete[] v.vecPtr;
    // Anzahl benutzter Elemente auf 0 setzen
    v.used = 0;
    // max. verfuegbare Elementanzahl auf 0 setzen
    v.maxmem = 0;
}

int main()
{
    // Objekt erstellen
    FloatVec fv;
    // Objekt initialisieren mit Reserve von 20 Plaetzen
    fv_init(fv,5);
    // Vektor mit 0.1 bis 1.7 fuellen
    for(float i=0.1; i<2.0; i+=0.4)
        fv_append(fv,i);
    // Vektor ausgeben
    fv_print(fv,5);
    // Vektor um einen Wert erweitern
    fv_append(fv,123.45);
    // Vektor ausgeben
    fv_print(fv,2);
    // Vektor löschen
    fv_delete(fv);
}

```