Informatik I (D-ITET)
Übungsstunde 8, 13.11.2017

Hossein Shafagh
shafagh@inf.ethz.ch
Problem 7.1. Reference Types

- a) T = int, S = int
  valid, valid during run-time, well-defined value: post: increment

- b) T = int, S = const int
  invalid: error: increment of read-only parameter 'I'

- c) T = int, S = int&
  valid, valid during run-time, well-defined value: post: increment

- d) T = int&, S = int
  valid, invalid during run-time: i is a local variable!

- e) T = const int&, S = int&
  valid, valid during run-time, post: increments the input variable and
  returns a reference to the input variable
**Problem 7.2. Array Declaration**

a) int array[4] = {1,2};
**Valid**! The elements of the array get the values 1, 2, 0, and 0 assigned.

b) float value[];
**Wrong**! A declaration without specifying the number of elements is not allowed.

c) double hello[] = {2.3, 5.3, 7.9};
**Valid**! The length is implicitly computed from the number of values.

d) unsigned int buffer[3] = {42, 507, -12, 2030};
**Wrong**! Four values can not be assigned to an array of length three.
Problem 7.3. Arrays in Practice

```cpp
int main() {
    const int LENGTH = 10;
    int values[LENGTH];

    // read ten values from input
    for(int i = 0; i < LENGTH; ++i) {
        std::cout << "enter integer " << i+1 << ":\n";
        std::cin >> values[i];
    }

    // compute the max and min
    int min = values[0];
    int max = values[0];
    for(int i = 1; i < LENGTH; ++i) {
        if(values[i] < min) {
            min = values[i];
        }
        if(values[i] > max) {
            max = values[i];
        }
    }
    std::cout << min << " / " << max << "\n";
    return 0;
}
```
Problem 7.3. Arrays in Practice

// compute the longest sequence of increasing numbers
int max_sequence = 1, count = 1;

for(int i = 1; i < LENGTH; ++i) {
    if(values[i] > values[i-1]) {
        count++;
        if(count > max_sequence) {
            max_sequence = count;
        }
    } else {
        count = 1;
    }
}
std::cout << "length of longest growing sequence: " << max_sequence << "\n";
Problem 7.3. Arrays in Practice

    // iterate and swap
    bool swapped;
    do {
        swapped = false;
        for(int i = 1; i < LENGTH; ++i) {
            if(values[i] > values[i-1]) {
                int temp = values[i];
                values[i] = values[i-1];
                values[i-1] = temp;
                swapped = true;
            }
        }
    } while(swapped);

    // output the array
    for(int i = 0; i < LENGTH; ++i) {
        std::cout << values[i] << " ";
    }
1 Sorting: bubble sort

https://en.wikipedia.org/wiki/Bubble_sort
2 Multidimensional Arrays/Vectors

```cpp
int array[4][3];
for (int i = 0; i < 4; i++)
    for (int j = 0; j < 3; j++)
        std::cin >> array[i][j];

int m = 4;
int n = 3;
std::vector< std::vector<int> > vector (m, std::vector<int>(n));
for (unsigned int i = 0; i < m; i++)
    for (unsigned int j = 0; j < n; j++)
        std::cin >> vector[i][j];
```
3 Representation of a Matrix in an Array
4 Strings

```cpp
std::string text;

std::cin >> text; // reads in a text of arbitrary length, for example "Hello"

text += " world!"; // appends text to the string, in this case changing it to "Hello world!

std::string text2 = text; // initialization also works with a string variable on the right hand side, in this case text2 = text

std::cout << text2 << "\n"; // outputs whole text stored in text2, here "Hello world!"
```
4 Strings

std::string str ("The quick brown fox jumps over the lazy dog.");

std::cout << str.find("fox") << "\n";       // outputs 16
std::cout << str.find("fox", 30) << "\n";   // outputs std::string::npos
    // (substring not found), a constant: -1

str.replace(10, 5, "red");
std::cout << str << "\n";                  // outputs "The quick red fox jumps over the
    // lazy dog."

str.erase(10,4);
std::cout << str << "\n";                  // outputs "The quick rex jumps over the lazy
    // dog."
5 Lindenmayer Systems and Turtle graphics
6 Pointers

```c
int i = 5;
int j = 7;
int& k = i; // k is initialized to reference i
k = j; // the value of i is changed to 7
```

- Change the value of the variable the pointer points to
- Change the pointer to have another target

```c
int a = 5;
int* x = &a;
```
7 Pointers on Arrays

- converting arr to an address implicitly
- Explicitly: \&arr[0] to get the same address directly
- pointer arithmetic
- dereference operator: *

```cpp
int arr[] = {7,1,0,2,5};

int* point = arr; // arr gets converted to the address of the first array element a[0]

std::cout << *point << "\n"; // outputs 7

std::cout << *(point + 3) << "\n"; // outputs 2
```
7 Pointers on Arrays

- Pointers and hence addresses behave similarly to unsigned int variables
- `ptr + 1`
- Pointer needs to know the length of the target (same type pointer!)
- For an `int` pointer, the length of the target is **4 bytes** (32 bit), so if one increments an int pointer by one (e.g., `++ptr`), 4 is added to the address saved in the pointer variable in order to point to the next element in the array.
- Incrementing a char pointer (e.g., `++ptr`) results in a shift of just one byte in order to point to the next element in the array.

```cpp
int arr[] = {7,1,0,2,5};

int* point = arr; // arr gets converted to the address of the first array element a[0]

std::cout << *point << "\n"; // outputs 7

std::cout << *(point + 3) << "\n"; // outputs 2
```
7 Pointers on Arrays

- pointer arithmetic
- past-the-end pointer

```cpp
int arr[] = {9,2,4,5,1,2,6};

for (int i = 0; i < 7; ++i)
    std::cout << arr[i] << "\n";

for (int* i = arr; i < arr + 7; ++i)
    std::cout << *i << "\n";
```
Problem 8.1. Vector and matrix multiplication

- Vector product (cross product)
- Matrix-vector product (column vector)
- Matrix product
- $3 \times 3$ matrices of type integer and 3 component vectors
- Arrays
- Tips: make separate functions to read/print vectors/matrix, ..
Problem 8.2. Decode binary NZZ front page

- On 8th June 2012, Neue Zürcher Zeitung went completely digital
- Each 8-bit binary number represented a single ASCII character (e.g., 01001110 01011010 01011010 encodes NZZ)
- `std::ifstream in(filename.c_str());` // open file
- `in >> value;` // read string from file
- `bool success = in >> value;` // true/false
Problem 8.3. Optional and for Fun: Lindenmayer systems

- We will show a selection of the submitted pictures in the lecture and on the homepage!
- Offline: image is saved as turtle_out.bmp in the same directory as your program is run from.
- Online: image is displayed in the output area of codeboard