UNIVERSITÄT DUISBURG ESSEN

NES Networked Embedded Systems

Programming in C/C++

Exercises 4

1. Calculate the maximum of two numbers in C++.
   - Write a preprocessor macro MAXM(). Use the most simple implementation that you can think of – it is intended that it behaves strangely in some situations…
   - Write a two overloaded functions maxf(), one accepting and returning int, the second accepting and returning double.

Test them with the following test program (can be downloaded as ex4task1.cpp):

```
int main(void) {
  int a = 10, b = 20;
  std::cout << "MAXM1 = " << MAXM(a,b) << std::endl;
  std::cout << "MAXM2 = " << MAXM(a,b+0.2) << std::endl;
  std::cout << "MAXM3 = " << MAXM(a,b++) << std::endl;
  std::cout << "maxf1 = " << maxf(a,b) << std::endl;
  // what's the problem with the following line?
  //std::cout << "maxf = " << maxf(a,b+0.2) << std::endl;
  std::cout << "maxf2 = " << maxf(a+0.1,b+0.2) << std::endl;
  std::cout << "maxf3 = " << maxf(a,b++) << std::endl;
  std::cout << "a = " << a << ", b = " << b << std::endl;
}
```

What are the advantages and disadvantages of the macro and function solutions? Why does the commented line in the code not compile? Which maxf() functions are called for the maxf1-maxf3 outputs and why? (code 10pts, comments 5pts, report 10pts)

2. Since the preprocessor basically replaces text, macros can also be used to generate code. Write a macro MAKEVAR that can be used like this

```
MAKEVAR(weight, int)
```

which is replaced by the preprocessor by

```
int weight;
void set_weight(int val) { weight = val };
int get_weight(void) { return weight; }
```

Add a comment to your macro to describe how it works.

Test your macro with the following test program (can be downloaded as ex4task2.cpp):

```
MAKEVAR(weight, int)
MAKEVAR(price, float)
int main(void) {
  set_weight(100);
  set_price(1.25);
  std::cout << "Weight: " << get_weight() << ", Price: " <<
get_price() << std::endl;
}
```

Note: To make long preprocessor statements more readable you can end a line with a backslash and continue it on the next line. For example

```
#define LONG_TEXT "This is a " \
"long text to be replaced"
```

(code 10pts, comment 5pts, no report)

3. What is the output of the following program (can be downloaded as ex4task3.c)? Describe how you use gdb to find the error but without single steps (commands “n(next)” or “s(tep)”)! Include all gdb comments in your report. Describe the error.

```
#include <stdio.h>

int main(void) {
  struct stest {
    int top;
    int arr[5];
    int bottom;
  } s;
  int i;

  s.top = 1;
  s.bottom = 2;

  for (i = 0; i <= 5; i++) {
    s.arr[i] = 42;
  }

  printf("top = %d, bottom = %d\n", s.top, s.bottom);
}
```

(report 20pts, no code and no comments)

4. Implement a class “StopWatch” that allows for measuring time. We will use this in future exercises.

- Method void start(void) starts the time measurement.
- Method double stop(void) stops the measurement returns the elapsed time since start.

Both methods shall print errors if they are not called in the correct order.
Test your class with the following program (can be downloaded as ex4task4.cpp):

```
int main(void) {
  StopWatch sw;
  sw.stop();
  sw.start();
  sw.start();
  for (int i=0; i <100000000; i++) ;
  std::cout << "Duration: " << sw.stop() << std::endl;
}
```

Hint: Look at the clock() function of header file “ctime”.
Note: The granularity of the clock() function is approx. 15ms. If you want to improve the precision, try clock_gettime() (in time.h) for Linux and cygwin or QueryPerformanceCounter() (in windows.h) for Windows instead. Look into the man pages or the MSDN. However, be aware the meaning of clock() is different than these time functions, but this does not matter in our case.
(code 35 pts, comments 5 pts, no report)