C and C++ Supervision Sheet 1

1. A function called “reverse” takes a string as a parameter, and reverses the order of its elements.
   a. Write an iterative implementation of reverse(). [2 marks]
   b. Write a recursive implementation of reverse(). [2 marks]
   c. Modify iterative reverse() so that it writes its output to the heap, returning a pointer to the new string. [2 mark]

2. For each of the following pointer expressions, explain what action it will perform (assume that p is a pointer to an int, q is a pointer to a char, and r is a pointer to a structure):
   a. p++                       f. *r->i
   b. q--                       g. *r->i++
   c. q = *p++                   h. (*r->i)++
   d. ++r->i                    i. *r++->i-- [10 marks]
   e. r++->i

3. Consider a preprocessor macro that exchanges its arguments.
   a. Define a macro SWAP(t, x, y) that exchanges two arguments of type t. [4 marks]
   b. Define a macro SWAP(x, y) that exchanges two arguments of the same type (e.g. int or char) without using a temporary variable. [5 marks]

4. Explain the difference between declaring and defining a variable. [4 marks]

5. A function needs to keep track of the number of times it has been called.
   a. Explain how the static keyword can be used for this purpose. [2 marks]
   b. How does static behave differently when used within the global scope? [2 marks]

6. Describe the layout of the memory components: Dynamic Memory Allocation, Data Segment, Code Segment and Stack. You may use an illustration as part of your explanation. [4 marks]

7. Most C compilers automatically define preprocessor macros that identify the platform on which the compiler is running (for example, Windows or Linux). This allows the programmer to include support for multiple platforms in a single source file, using conditional compilation. Write a program that prints “Hello, world in <platform>”, that illustrates this concept (and state your assumptions). [4 marks]

8. Explain why the code shown below, when executed, will print the value 420. [4 marks]
```c
#include<stdio.h>
#define init_employee(X,Y) {(X),(Y),wage_emp}
typedef struct Employee Em;
struct Employee {int hours,salary;int (*wage)(Em*)};
int wage_emp(Em *ths) {return ths->hours*ths->salary;}
#define init_manager(X,Y,Z) {(X),(Y),wage_man,(Z)}
typedef struct Manager Mn;
struct Manager {int hours,salary;int (*wage)(Mn*)};
int wage_man(Mn *ths) {return ths->hours*ths->salary+ths->bonus;}
int main(void) {
    Mn m = init_manager(40,10,20);
    Em *e= (Em *) &m;
    printf("%d\n",e->wage(e));
    return 0;
}
```