Page 1112#1-4

19.1 What happens if a recursive function never returns?

Literally speaking, the ESI/RSI pointers will add up on the stack until the stack (which defaults to 2040KB on most platforms that I'm aware of) runs out of room. The stack explodes, the program crashes (segmentation fault). I suppose that on some interpreters without a call stack size limit it could loop forever until the external stack information grew larger than the available memory, at which point it would explode, the program will crash.

19.2 What is a recursive function's base case?

The base case is a code path that does not include recursion (calling itself again).

19.3 What will the following program display?

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

// Function prototype
void showMe(int arg);

int main()
{
     int num = 0;
     showMe(num);
     return 0;
}

void showMe(int arg)
{
     if (arg < 10)
          showMe(++arg);
     else
          cout << arg << endl;
}
```

```
10.
```

19.4 What is the difference between direct and indirect recursion?

In direct recursion, a function will call itself. In indirect recursion, a function may be called again in a code path which may descend through one or more other different functions.

Page 1133#10-14

10 Write a recursive function to return the number of times a specified number occurs in an array.

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

// roughly inspired by strtok_r
template<class T>
size_t arr_count_r(T * haystack, T needle, size_t
haystack_size, size_t search_start = 0) {
  if (search_start == haystack_size - 1) {
    return haystack[search_start] == needle;
  } else {
    return (haystack[search_start] == needle) +
            arr_count_r(haystack,
                        needle,
                        haystack_size,
                        search_start+1);
  }
}

int main(int argc, char *argv[]) {
  int toxic_farts[] = { 1, 2, 3, 2, 4, 7, 9 };

  int needle = 3;
  cout << "number of times " << needle
       << " appears in toxic_farts: "
       << arr_count_r<int>(toxic_farts,
                           needle,
                           sizeof(toxic_farts)/sizeof(int))
       << endl;

  return 0;
}
```

11 Write a recursive function to return the largest value in an array.

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

#define HAMBURGER_MUSIC 0
```

```
template<class T>
T find_obesity(T * haystack,
    size_t haystack_weight,
    size_t cursor = 0) {
  if (cursor == haystack_weight - 1) {
    return haystack[cursor];
  } else {
    T next_contender = find_obesity(haystack,
                          haystack_weight, cursor+1);
    return (haystack[cursor] > next_contender) ?
              haystack[cursor] : next_contender;
  }
}

int main(int argc, char *argv[]) {
  int america[] = {
    INT_MAX - 12,
    INT_MAX - 13,
    INT_MAX - 5,
    INT_MAX - 20 };

  cout  << "the fattest integer in america is: "
        << find_obesity<int>(america, sizeof(america)/
              sizeof(int))
        << endl;

  // [clapping intensifies]

  return HAMBURGER_MUSIC;
}
```

What is output of following programmes?

12 55

13

```
**********
*********
********
*******
******
*****
****
***
**
*
```

14

Hello
olleH