Exercise #8 – Pointers
Due 18/2/99

Exercise 1 (dissection)

Dissect the following program (i.e. explain how it runs, step by step):

```
#include "iostream.h"
#include "stdlib.h"
#include "time.h"

void print_array(int *a, int size)
{
      int i;
      cout << "a = [";
      for (i=0; i<size-1; i++)
            cout << a[i] << ',';
      cout << a[i] << "]\n";
}

int sum_array(int *a, int size)
{
      int i, sum=0;
      for (i=0; i<size; i++)
            sum += a[i];

      return sum;
}

void fill_array(int *a, int size)
{
      int i;

      for (i=0; i<size; i++)
            a[i] = rand() % 19 - 9;
}

void main()
{
      int *a, n;

      for ( ; ; ) {
            cout << "Input n: ";
            cin >> n;
            if (n < 1) {
                  cout << "Bye Bye\n";
                  return;
            }
            cout << endl;
            a = (int *) calloc(n,sizeof(int));
            if (a == NULL)
                  return;
            fill_array(a,n);
            print_array(a,n);
            cout << "Sum of elements = " << sum_array(a,n) << "\n\n";
            free(a);
      }
}
```

Exercise 2 (LARGE numbers – cont.)

Modify the LARGE-NUMBERS program to work with arrays of varying sizes:

1. Represent a LARGE number by an array whose last element is –1 (signaling end-of-number).
2. Write a function `int length(int *a)` that computes the number of digits in a large-number.
3. Write a function `int* add(int *a, int *b)` that finds out the lengths of `a` and `b`, allocates enough memory to contain the result, and adds the two numbers (represented in `a` and `b`) into that memory.
4. Write another function `int* subtract(int *a, int *b)` that finds the lengths of `a` and `b`, allocates enough memory to contain the result, and returns the value of `|a - b|`. Your function may use the following steps:
   (a) Write a comparison function `int larger(int *a, int *b)`, returning 0 if a ≤ b and non-zero otherwise.
   (b) Now subtract the smaller number from the larger number.

5. *(easy bonus) Write a multiplication function.
6. Write a main program that tests your functions, filling the arrays with random numbers.