Assignment 5 - Software I, Spring 2003 (0368-2157-9,0368-2157-12)

http://www.cs.tau.ac.il/~efif/courses/software1

Due: May 20, 2003

In addition to the standard guidelines, which you should know by heart now, make sure your code is comprehendable. Comment the code and use meaningfull names for the variables and the functions.

Ex 5 big_factorial

Write a program that computes the arbitrary-large factorial of an integer. The program reads an integer $n < 2^{32}$ from the command line, computes its arbitrary-large factorial, and prints it out.
You need to implement a data structure, namely **Big_int**, that represents an arbitrary-large non-negative integer, and a few operations on this data structure listed below.
Download the header file `Big_int.h` from
`http://www.cs.tau.ac.il/~efif/courses/software1/code/big_factorial/Big_int.h`. This file contains the following:

Defines

```
#define BASE 10
```

A directive that defines the base of the format of the Big_int.

Data Structures

```
typedef struct digit {
  unsigned char d;     /* A single digit in BASE format */
  struct digit * next; /* A pointer to the next digit in the list */
} Digit;
```

An element in a linked list of digits. This struct is used by the Big_int structure. The digits in such a list represents an arbitrary large integer in BASE format, where the first digit in the list is the least significant and the last digit is the most significant.

```
typedef struct big_int {
  int num_digits;      /* The number of digits in the list of digits */
  Digit * digits;      /* A linked list of digits */
} Big_int;
```

A struct that represents an arbitrary large integer.

Operations

`extern void bi_init(Big_int * bi, unsigned char d);`

Initialize a Big_int to a number represented by a single digit

`extern void bi_mul(Big_int * res, Big_int * a, unsigned int b);`

Assign a Big_int with the product of a Big_int and an **unsigned int**.

`extern void bi_print(Big_int * bi);`

Print a Big_int to the standard output

`extern void bi_clean(Big_int * bi);`

Clean a Big_int. Deallocate all memory allocated for internal use

The comments in the file contain keywords that can be interpreted by the Doxygen documentation system. Visit the page `http://www.cs.tau.ac.il/~efif/courses/software1/code/big_factorial/html` to see the documentation produced by doxygen.

Files Names

As usual place the files for the assignment under `~/software1/assign5`. This time, you need to put the implementation of the operations listed above in the file `Big_int.c` and the `main()` function in a separate file `big_factorial.c`. You need to provide an appropriate makefile file, that can be used to build the executable `big_factorial`. Note that names are case sensitive (i.e. `Big_int.c` is different than `big_int.c`).

Examples

Command:

```
big_factorial 50
```

Output:

```
30414093201713378043612608166064768844377641568960512000000000000
```

Good Luck!

More Information on the Submission

Giving Permission to the Files

Before submitting the solution set, please give permission to the files by executing the following command:

```
chmod 705 ~ ~/software1 ~/software1/assign5 ~/software1/assign5/*
```