Duration: **1 hour and 45 minutes**
Aids Allowed: **Reference sheet handed out with the test only**

Student Number: |_|_|_|_|_|_|_|_|_|_|_|_|_|

UTORid: _____

Family Name(s): _____

Given Name(s): _____

UofT email: _____@mail.utoronto.ca

---

*Do **not** turn this page until you have received the signal to start.*
In the meantime, please read the instructions below *carefully*.

This midterm test consists of 5 questions on 16 pages (including this one), printed on both sides of the paper. *When you receive the signal to start, please make sure that your copy of the test is complete, fill in the identification section above, and write your name on the back of the last page.*

You **must write your student number at the bottom of every odd-numbered page**. **Failure to do so will result in a deduction of 5 points.**

Answer each question directly on the test paper, in the space provided. *indicate clearly the part of your work that should be marked.*

If you need extra space, you can use the empty pages at the end of the midterm booklet. **Indicate clearly that you have done so if you have**.

When you are asked to write code, *no* error checking is required: you may assume that all user input and argument values are valid, except where explicitly indicated otherwise.

Write up your solutions carefully! Comments are *not* required to receive full marks, except where explicitly indicated otherwise. However, they may help us mark your answers, and part marks *might* be given for partial solutions with comments clearly indicating what the missing parts should accomplish.

You may use functions written to answer one question as part of the solution to another question.

Write your answers in C. You may `#include` any header file that is available with a standard installation of `gcc` or is described in the C99 standard, **unless otherwise specified**. We will accept code that is correct under the C99 standard as well as any code that will run correctly when compiled with `gcc`.

For full credit, your code should run correctly without crashing. We will not deduct points for memory leaks.

MARKING GUIDE

1: _____/15

2: _____/15

3: _____/20

4: _____/10

5: _____/30

TOTAL: _____/90

Good Luck!

OVER...

Question 1. [15 MARKS]

Write a function that appends a node to a linked list. You must use the definitions below. Make sure that the code works on lists of **any size**. If you need helper functions, you must write them yourself.

You may need to make assumptions like "the next field of the last node is NULL." State any such assumption as a comment.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

typedef struct node{
      int data;
      struct node *next;
} node;


typedef struct LL{
      node *head;
      int size;
} LL;



// append integer new_elem to linked list my_list
void LL_append(LL *my_list, int new_elem)
{
```


CONT'D...

Question 2. [15 MARKS]

Write a function with the signature `char * strcat_rec(char *dest, const char *src)` that concatenates `dest` and `src` recursively. The result is to be stored at `dest`. The function returns the destination string `dest`. The function does the same thing as `strcat` in `string.h`. You **must use recursion**. You **must not** include header files, use helper functions, or use loops. You may assume that there is sufficient space at `dest` to store the resultant string.

```
char *strcat_rec(char *dest, const char *src)
{
```

Question 3. [20 MARKS]

You have been tasked with creating a digital White Pages book for the Province of Ontario. You have been provided with a text file. The first line of the file contains the number of entries present. Each of the following lines contains a phone number formatted as XXX-XXX-XXXX, followed by a tab, followed by the business name.

```
3
226-222-6678    Spring Cleaning 4 U
289-778-1123    Nowruz Party Planners
416-445-2555    The April Shower Curtain Company
```

Part (a) [15 MARKS]

Write a function to read the file `filename` and set the values at the input pointers appropriately so that the information can be used outside the function. The data must be stored as a contiguous block of `struct business`'s.

Assume memory allocated in this function will be freed elsewhere. Assume that the number of businesses can fit in an `int`. You **may not** assume that you know the maximum size of a business name. Your code needs to be able to handle business names of any length.

We strongly recommend, but don't require, using a `while`-loop rather than `strtok` or `sscanf` in order to read in the phone numbers and business names. It is MUCH easier to use a `while`-loop!

```
typedef struct business {
  int phone_number[10]; // Note: this is an array of 10 integers
                        // (one per digit), not a character array
  char *name;
} business;

void read_numbers(const char *filename, business **whitepages, int *size)
{
```

Part (b) [5 MARKS]

Write a function with the signature `void free_numbers(business *whitepages)` that appropriately frees all the memory allocated by the function `read_numbers`. Assume that `whitepages` was set by `read_numbers`, and that `size` is the number of items that were read in.

```
void free_numbers(business *whitepages, int size)
{
```

Question 4. [10 MARKS]

Consider the following `struct`

```
typedef struct daylight{
    int hours;
    int minutes;
} daylight;
```

Part (a) [5 MARKS]

Complete the following function, which creates a block of memory that can contain **n/2 pointers** to `struct daylight`, and then sets the data at those pointers to the following:

```
first pointer: hours = data[0], minutes = data[1]
second pointer: hours = data[2], minutes = data[3]
...
n/2-th pointer: hours = data[n-2], minutes = data[n-1]
```

The function must be usable in later Parts of the question. The block must store **pointers** to `struct daylight`, not `struct daylight`s.

```
void create_daylight_pt_block(                          , int *data, int n)
{
```

Part (b) [2 MARKS]

On March 7, 2023, the day length in Toronto is 11 hours and 29 minutes. During the solar equinox on March 20, 2023, the day length in Toronto will be 12 hours and 9 minutes.

Write code which uses `create_daylight_pt_block` to store the day lengths of the two days. The data in the created structure must be accessible for later use.

Part (c) [3 MARKS]

Write code that prints out the number of hours in the second (i.e., at index 1) element of the block.

Question 5. [30 MARKS]

Consider the following Abstract Data Type: a `pyinteger` is an integer that can be of unlimited length (i.e., no limit on the number of digits; you **must** consider this fact when answering this question). The integer supports the `plusplus` and `addition` operations:

`plusplus(n1)` adds 1 to the `pyinteger n1`

`add(n1, n2)` adds the `pyinteger n2` to the `pyinteger n1`

Both operations modify the contents of the `pyinteger n1`.

You may find the following a useful reminder of how long addition works.

```
    1 1              1
  1 0 0 9 9        6 7
+         1    +   8 9
-----------      -----
  1 0 1 0 0        1 5 6
```

Part (a) [10 MARKS]

Implement the data structure `pyinteger`, along with the operation `plusplus` (N.B.: `add` will be implemented in Part (c)). There must be an `h` file that one can include, as well as a `c` file with the implementation of the necessary functions. In your answer, indicate which code is in which file. It must be the case that the user can include the `h` file and then be able to compile their program with an alternative data structure for `pyinteger`. **The other implementation details are up to you.**

CONT'D...

Part (b) [10 MARKS]

Write file `main.c` that creates a `pyinteger` and then uses the operation `plusplus` and displays the new value of the `pyinteger` that was modified. For example, if the `pyinteger` started as 999, your code should modify the `pyinteger` and print the new value of the `pyinteger`, which is 1000.

Part (c) [10 MARKS]

Implement the operation `add`. Indicate tight upper bound on the time complexity of your implementation using big-Oh notation, in terms of n, where n is the magnitude of the value stored in the `pyinteger`. Briefly explain (a few sentences is enough) why your complexity estimate is correct. Make your implementation as efficient as possible in terms of asymptotic time complexity. Inefficient but correct solutions will be penalized by 2 points. The time complexity estimate + correct explanation is worth 3 points. For this Part, you only need to write code that would go in the `c` file.

CONT'D...

On this page, please write nothing except your name.

Family Name(s): _____

Given Name(s): _____