

**CSC 357: Systems Programming**

Midterm

* Put your name on all sheets.
* No books, notes or other type of written material are allowed.
* Mobile phones are not allowed.
* Show your work neatly for partial credit.
* You must show your work to receive any credit.
* Max points: 100 + 4 bonus.
* You can use the backside of the sheets.
* Do not get distracted by answering unnecessary questions.
* Read the statement below and sign your name.
* Use your laptop. You are not allowed to use anything else than your IDE except:

<http://man7.org/linux/man-pages/man2/mmap.2.html> and <https://linux.die.net/man/2/munmap>

I affirm that I neither will give nor receive unauthorized assistance on this examination. All the work that appears on the following pages is entirely my own.

Name:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

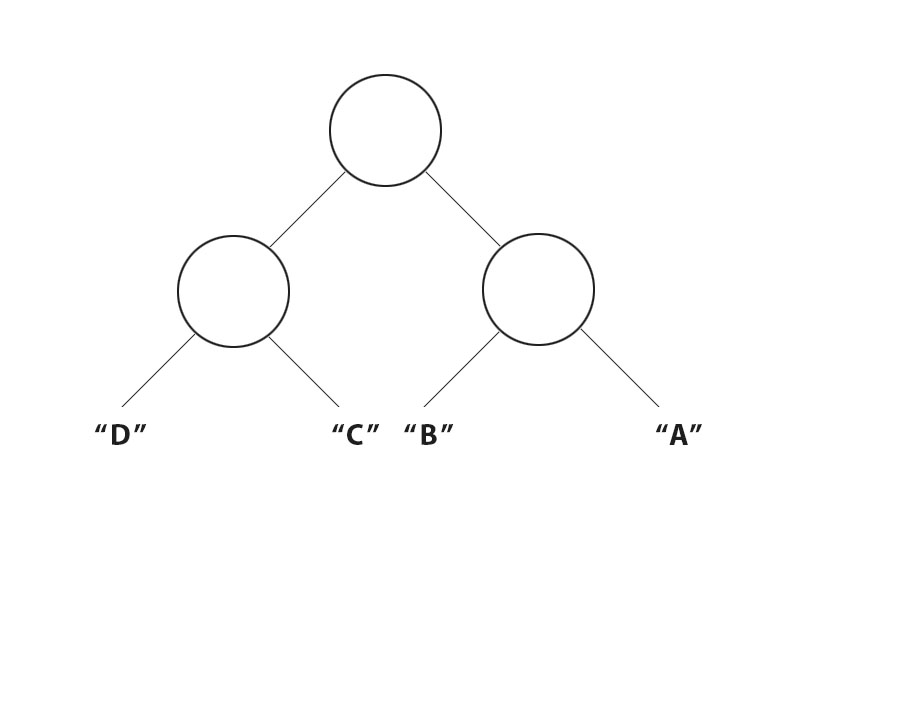
Signature:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| Total points | Letter grade |

**Task 1: forking**

Print the numbers in the order seen on the image. Use fork() and wait() to start new processes in 100% accordance to the image. You need to decide where child- and parent processes are. (Sorry for the bad line drawing, they will be better on the real midterm ;)



**Task 2: lists**

Program a double linked list from scratch. Start with an empty list head, which should be global. The data segment is an integer. Implement following functions:

new\_element() //should read an integer, malloc a new element and puts it at the front (!) of the list

printlist()//prints the whole list (data segment and addresses prev & next)

remove\_element(int d)//finds all elements with data “d” and removes them from the list (free).

delete\_list()// deletes (free) the whole list)

Use following structure:

typedef struct listelement

{

struct listelement \*prev,\*next;

int number;

}listelement;

**Task 3: read from bitmap**

Open a bitmapfile (the two fileheader/fileinfoheader structures will be provided) and permutate the colors in following way: b,g,r -> r,b,g

**Task 4: read from binary file**

Open a binary file “file.bin”. The content of the file is following structure:

typedef struct datafile

{

char type; //

int size;//size of the data part in bytes

}datafile;

Make an instance of datafile in your main function. Read from the file into this structure (its up to you how). After the struct, the file contains the data part with the size given in the structure. Allocate enough mem and read from the file!

Now check the “type”.

If type is “U”, loop through your data array and convert all ASCII signs following way:

a[i] = a[i] + 4;

“T”

a[i] = a[i] - 2;

“M”

a[i] = a[i] + 9;

Do this for all signs except for the value 10 (new line) and 0 (end of string)

Save the whole resulting text in a text file (without the struct at the beginning).

**Task 5: memory allocation**

Write a program, that reads text from the keyboard. One word is fine, no need for whole sentences. Allocate exactly what you need for each text on the heap with mmap().

If you enter the same text again, remove this word from the heap with munmap().

If you enter “print”, write all words entered so far.

Maximum word count is 10, so you don’t have to (but you can) use a list. An array like:

char \*text[10]; will do too. Be careful not to print deleted words again!

**Task 6: multi-tasking**

Parallel efficiency!

Have an array “arr” of 1.000.000 elements (gonna approach that dynamically, wouldn’t be mentioned on the midterm. Free it properly after use or points minus). Fork in a way to achieve 2 processes which solve for each element:

arr[i] = i\*5 + i/2;

Measure the time with clock\_t stop,start = clock(); and print the time.

Now repeat that experiment with 4 processes (in the same program). Make sure no oder child process is still running when starting this. Print the time. After time measuring, print the array. Lets just make sure the child processes had access to the same array!

**Task 7: answer the 20 questions on PolyLearn**

The 20 questions will be from the pool of all lecture quizzes so far plus the Monday questions before the midterm.