

University of Ontario Institute of Technology (UOIT)

Faculty of Engineering and Applied Science (FEAS)

Department Of Electrical, Computer, And Software Engineering (ECSE)

**Operating Systems (SOFE 3950)**

Tutorial #3 Activity | Submission

Yong Deng

Winter 2020

Group Members

Aryan Kukreja (100651838)

Emil Ilnicki (100659072)

Sydney Smith (100654205)

Harasees Singh Gill (100656810)

# Conceptual Questions

## Question 1

The **fopen()** function has the following attributes.

1. The **“r”** mode opens the file in *read mode*. This is for reading the file. The file passed in must exist.
2. The **“w”** mode opens the file in *write mode*. If the file exists, its contents are erased; else a new file is created.
3. The **“r+”** and **“w+”** are 2 ways to open the file in *read and write mode*. In **r+**, the file must already exist. In **w+**, a new empty file is created.
4. The **“a”** mode opens the file in *append* *mode*. This mode opens a file for writing. The file’s contents are maintained, and new data is added at its end. The file is created if it does not exist.

## Question 2

A **heap** is used for dynamic memory allocation. It is stored in the RAM of the computer. Heap memory is allotted at runtime, and its access is a bit slower, but the heap size is significantly larger, only restricted by the size of the virtual memory.

Key differences between a stack and a heap are:

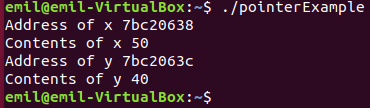
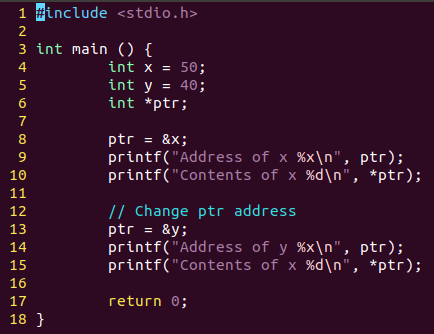
|  |  |
| --- | --- |
| **Stack** | **Heap** |
| This data structure is linear | This data structure is hierarchical |
| Super-quick access in memory | Somewhat slower to access than a stack |
| Efficient space management (prevents fragmentation) | Inefficient use of memory can often lead to fragmentation in the memory |
| Only use local variables, and cannot resize them | Can use global variables and can resize variables |
| Stack size is OS-dependant | No limit on the size |
| Memory allocation is a set of consecutive blocks | Memory is allocated randomly |
| Automatic allocation/deallocation | Manual allocation/deallocation |
| Cheaper | More expensive |

## 

## 

## Question 3

Pointers are variables that point to an address in memory, these addresses can hold some information such as a character, integer etc.

pr

## Question 4

Malloc is used to allocate memory space as a heap for a given size of bytes. If it can successfully allocate the space, it returns a pointer of type void that can then be type casted into any type of pointer, but if it cannot allocate the space, it returns a pointer of type NULL.

To use Malloc successfully first declare a pointer to an empty variable, then you set that variable to be an empty block of memory using malloc(size\_t *size*), and by keeping in mind that malloc returns a void pointer, in that same variable declaration you can add the type of pointer you want the variable to become(i.e. (int\*)malloc(size\_t *size*)).

Malloc stores data in heaps. If it remains allocated when out of use it can result in memory leaks which can then result in system crashes. Once a user is done using the memory they allocated, it is vital to free that memory using free().

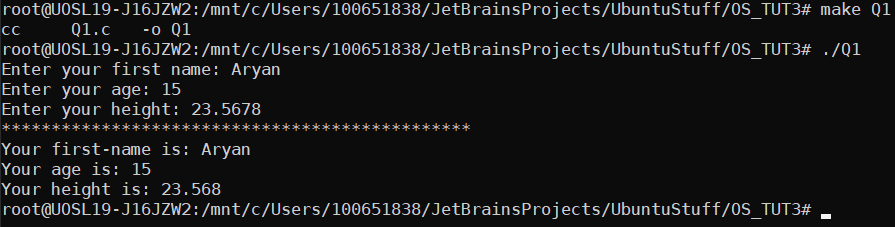
## Question 5

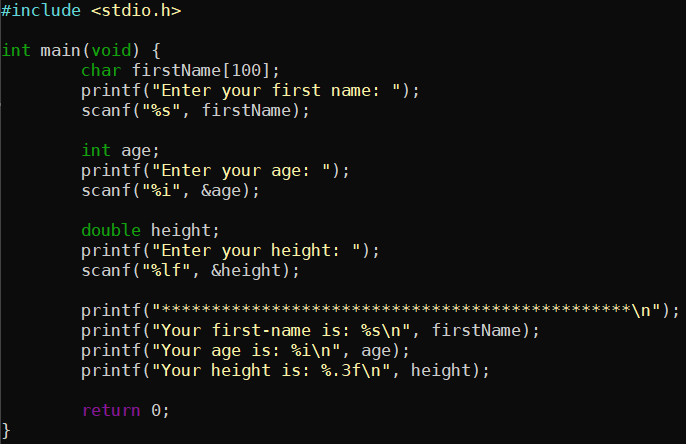
Malloc and calloc both allocate blocks of memory but the major difference between them is that malloc does not initialize that block of memory to 0 and calloc does. This can be offset by using memset(void \*str, int c, size\_t n) following malloc where \*str is a pointer to the empty block of memory allocated by malloc, c is the value to initialize with, and n is the size in bytes of the space to be initialized.

To do this same thing, calloc takes in one more parameter than malloc does. calloc(size\_t *nitems*, size\_t *size*) where nitems is the number of elements to be allocated, and size is the size of each element to be allocated.

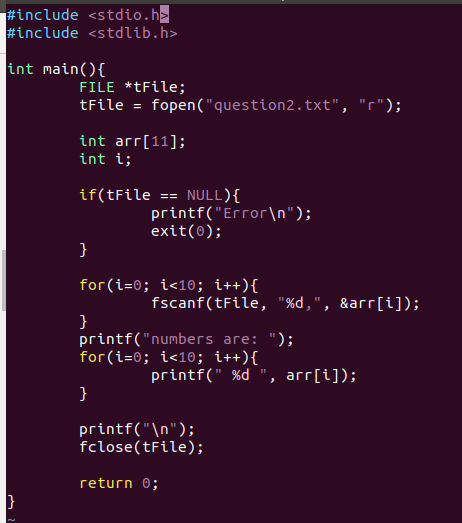
# Application Questions

## Question 1



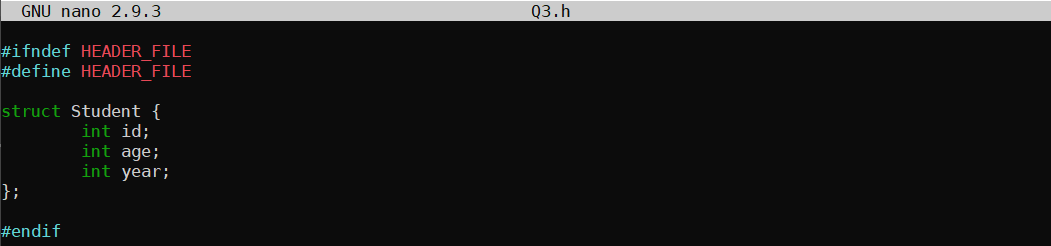
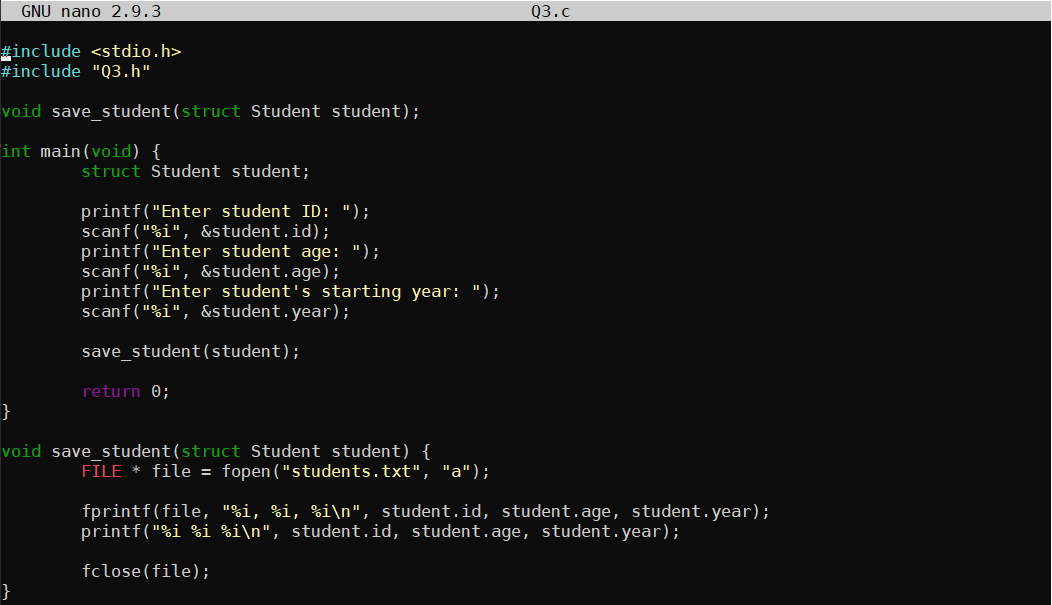
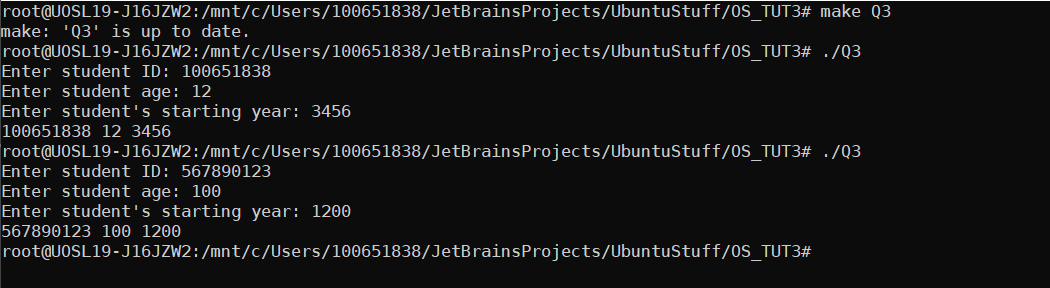


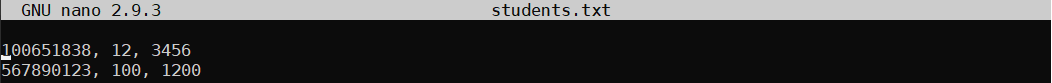
## Question 2



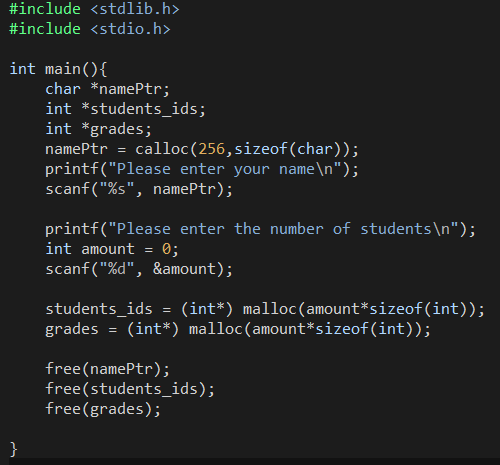


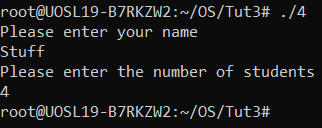
## Question 3





## Question 4





## Question 5

