

Tutorial 3

Student Names:

Aaron Spindler, Shomari Simpson, Joseph Robertson

Conceptual Questions

1. List each of the modes for the fopen function to perform the following operations: read, write, read and write, append to a file.

R - read

W - write

R+ - read and write

A - append

2. Does dynamic memory use the stack or heap? What is the difference between the stack and heap?

Heap is used for dynamic memory and stack is used for static memory. The stack trivially allocates and deallocates memory while the heap has more complex bookkeeping for allocation.

3. Explain what a pointer is, and provide examples (in C code) of how to change the address that a pointer points to and how to access the data the pointer points to.

A pointer is a variable which contains the address in memory of another variable.

`int *ip //pointer variable declaration`

`printf ("%d", *ip) //access the value using the pointer`

4. Read the documentation on the malloc and free functions and explain briefly how to use malloc.

Malloc allocates a block of size bytes of memory, returning a pointer to the beginning of the block

5. What is the difference between malloc and calloc?

Calloc allocates a block of memory for an array of num elements, each of them size bytes long, and initializes all its bits to zero. Malloc's newly allocated block of memory is not initialized so the values are not determined.

1. Create a program that does the following

- Prompts the user for their first name, age, and height (hint use a character array for strings).
- Prints back to the console, their first name, age, and height
- You will need to review the scanf documentation to complete this

2. Create a program that does the following

- Reads the ten integers from the included file question2.txt
- Stores each integer read from the file in an array

- Prints the contents of the array to the terminal
- You will need to review the fopen and fscanf documentation

3. Create a program that does the following

- Create a struct called student containing their student id, age, and the year they started at UOIT.
- Create a function called save_student which does the following
 - Takes as its argument the student struct and returns void
 - Opens a file called students.txt in append mode
 - Saves the student id, age, and year from the students struct to the file on one line delimited by commas (e.g. 100123456,19,2014).
- In the console prompt the user for their student id ,age , and the year they start at UOIT.
- Store the values entered by the user in the student struct.
- Call the function save_student with the student struct to save the data to the students.txt file.

4. Create a program that does the following

- Creates three pointers, a character pointer professor, and two integer pointers student_ids, grades
- Using dynamic memory, use calloc to allocate 256 characters for the professor pointer
- Prompts the professor for their name , and the number of students to mark.
- Stores the professor's name using the professor pointer and in an integer the number of students to mark.
- Using dynamic memory, use malloc to allocate memory for student_ids and grades to hold the number of students the professor needs to mark.
- The program does not need to do anything else, ensure that you free your memory before terminating.
- You will need to review the malloc, calloc, and sizeof documentation.

5.

Building upon the previous questions you will create a marking system for professors at UOIT.

- Structs can be used the same as any other data type in C, instead of having two arrays for the grades and student ids create a struct called grade that contains two integers: student_id and mark.
- Create a function grade_students which takes the following arguments: a pointer to the grade struct called grades , and an integer num_students. The function returns void and does the following:
 - Opens the file grades.txt in write mode
 - Using the num_students parameter iterates through all of the grade structs pointed to by the grades parameter (remember arrays are pointers, you can treat pointers like arrays).
 - For each grade structure adds the mark member of the struct to a variable called sum that holds the sum of all student's grades.
 - For each grade structure write to the file grades.txt the student id and the mark on a single line.
 - After adding every student's mark to the sum variable, calculate the average (mean) and standard deviation , you will need to use <math.h> don't forget when you compile to add -lm
 - Write to the file grades.txt the average and standard deviation that you calculated.
 - Create two pointers, a character pointer professor , and a pointer for the grade struct you created and call it grades , it will hold an array of grade structure