Assignment 1

COMP 2401

Date: September 20, 2021

Due: on October 3, 2021 by 23:00

Submission: Electronic submission on Brightspace

- **Submission of Part I - Please type the answers and then submit the pdf file.**
- **Submission of Part II**
  - Submission must be in Brightspace
  - Submit a single tar file with all the c and h files that you created. The file name is a1.tar

Do Not post the assignment or the assignment and\or solutions on any website

Objectives:

1. Understanding number representation of integers and floats
2. Coding a program in C:
   2.1. Using arrays
   2.2. Using call by value and call by reference
   2.3. Determining function parameters and types
3. Using a debugger is optional

Grading:

Maximum grade for the assignment is 100

Part I – 80 points

Part II - 30 points

Bonus 10 points

Note– 8 points per question (1-10). Question 11 is a bonus question. Note that the total includes 10 discretionary points for submitting clean and readable assignment. It is up to the TA to deduct points for improper submission.

Copyright

Part I Numbers Representation

Maximum grade for this section is 80 – 8 points per question (1-10). Question 11 is a bonus question. Note that the total includes 10 discretionary points for submitting clean and readable assignment. It is up to the TA to deduct points for improper submission

Show your work.

1) (6 pts.) How many different values can be represented using 9 digits in the following bases? Justify your answer.
   a) Base 2 (binary numbers)
   b) Base 16 (hexadecimal numbers)
   c) Base 8 (octal numbers)

2) (8 pts.) Find the decimal equivalent of the following:
   (a) $(542)_8$
   (b) $(EAB1)_{16}$
   (c) $(2E)_{16}$
   (d) $(00010111)_2$ - number is one byte long

3) (9 pts.) What is the simple binary, octal and hexa equivalent of following decimal numbers:
   (a) 231 (b) 1183 (c) 1928
   Show your work. Note, that it is sufficient to solve it using one of the bases and then convert the representation from one base to another.

4) (6 pts.) Assume that a system has 6-bits bytes for storing binary numbers. Specify the range of numbers that can be represented in one byte in the following data models:
   a) Unsigned Integers (i.e., only positive numbers) -
   b) 2's Complement

5) (9 pts.) Perform the operation on the following 8-bit binary numbers. The numbers are represented in 2's-complement notation. Show the work using base 2.
   a) 01011101 + 10101001
   b) 10110111 - 11001011

Copyright

c) 00000111 * 00000101

6) (6 pts.) Convert the following decimal numbers into 2's complement representations. Assume that 8 bits are used to store the numbers.

a) 72

b) 0

c) -128

d) -5

7) (10 pts.) Find the decimal equivalent of the following binary numbers assuming that each number is expressed in:
(1) unsigned integer (2) 2's complement (3) Excess 127

(a) 01011000 (b) 10101001

8) (6 pts.) Convert the following decimal numbers to normalized floating point representation (e.g., 0.5 should be 1.0×2^{-1}) Show your work

(a) 4.625 (b) -1.25

9) (6 pts.) Convert the following binary integers directly into (i) octal (ii) hexadecimal, (by grouping the bits)
(a) 101111110010 (b) 000101111110

10) (14 pts.) Assuming that you are required to make an 8-bit floating point representation, with a 3 bits exponent using excess-4 notation and a 4 bits mantissa using normalized notation.

a) How would the number 0.5 be stored in the byte?

b) What is the number that its binary representation is 11101010?

c) Can the number 8.75 be stored in this system? If it can be stored then show the bit pattern, if it cannot be stored explain why it cannot be stored?

d) Can the number 6.25 be stored in this system? If it can be stored then show the bit pattern, if it cannot be stored explain why it cannot be stored?

11) (Bonus question 10 points) Assuming that you are required to make an 8-bit floating point representation, with a 3 bits exponent using excess-4 notation and a 4 bits mantissa using

Copyright

normalized notation. What is the **smallest positive** number that can be stored (show the bit pattern)? What is the **smallest** negative number that can be stored (show the bit pattern)?

Part II Simple Coding

Maximum grade for this section is 30

Grading:

1) Function populateArray() – 10 pts.
2) Function printArray() – 4 pts.
3) Function removeValues() – 12 pts.
4) Compute GPA and printing – 4 pts.

Note that grades include commenting, no usage of global variable, clear and simple code, and no code repetitions (if needed add functions to avoid repeating code sections).

Coding instructions:

1) Add functions as needed.
2) Code must compile and execute on the course VM
3) Use the file arrays.c. Do not change any of the function names or the file name.
4) Available files:
   a) a1_source_files.tar – the tar file consists of the template file arrays.c and an executable file arrays_exec which you can experiment with.

      Note, that once you started the program you need to enter the grades. The program accepts grades until a -1 was entered or 20 grades were entered. Also, the program does not check for input for validity.

Suggestions:

Write the design of the code within the function body and then comment it. This way you will catch two birds in once step: a. the function will already include comments and b. you will only need to complete a few lines of code under each comment. See function main() as an example.

In this section you are asked to write a very short program to compute the average GPA of a student. Here you will need to add code to three functions – populateArray(), printArray(), and removeValues(). Note that you can add additional functions as you see fit.

Once the grades were entered the program needs to (follow the flow in the function main():

1) Print the grades in one row
2) Compute the average gpa of the grades
3) Print the number of courses and the gpa
4) Remove all grades that are lower than 7 from the array

Copyright

5) Print the remaining grades
6) Compute the average gap of the remaining grades
7) Print the number of courses and the average gpa
8) Compute the

The following functions must be coded. Note that you can code additional functions as needed.

1) Function populateArray() – this function reads the grades until either a grade of -1 is given as input or the maximum size of the array was reached. There are no restrictions on how the grades can be entered, i.e., the grades can be entered one at a time, several in a row, or several in multiple rows. Add comments as needed
   Note that here parameters of the function are provided.

2) Function removeValues() – here the function needs to accept a value (e.g., minValue) and then remove all values in the array that are smaller than minValue. For example, if minValue is 7 and the array has the elements 3 6 8 9 2 12 then elements 3, 6 and 2 should be removed and the array and the array should only consist of 3 elements – 9, 8 and 12. Note that order of grades does not need to be preserved.

   Here you will need to determine the function prototype (i.e., the parameters list) as well as add comments to the function (i.e., comments in the code and the parameters input and output).

3) Function printArray() – here the function needs to print the grades in one line with a single space between the grades. Once the grades were printed in a row then the function needs to move to the next line. Comments about the function were already added. Add comm
   Comments within the function were not added.

Example of execution:

student> a.out

3 4 5 7 9

8 -1

The courses grades are:

3 4 5 7 9 8

The number of courses = 6 average GPA = 6.00

Removing all gpa elements smaller than 7

The grades with grades >= 7 are:

8 9 7

The number of courses = 3 average GPA = 8.00

Copyright

Another example

```
student> a.out
3 5 9 8 -1
The courses grades are:
3 5 9 8
The number of courses = 4 average GPA =  6.25
Removing all gpa elements smaller than 7
The grades with grades >= 7 are:
8 9
The number of courses = 2 average GPA =  8.50
```

Copyright