Exam preparation questions

This document contains examples of programming questions that can help you to prepare for the exam. The questions on the actual exam will be different, but if you can solve the problems here, and you have done the work packages, you should have no problem with the exam.

C Programming

I hope that you remember how to add fractions:

$$\frac{a}{b} + \frac{c}{d} = \frac{ad + bc}{cd}$$

Usually, the result is simplified by dividing the numerator and denumerator by the greatest common divisor.

a) Write a function

```
int addFraction(int a, int c, int b, int d)
```

which prints the sum of a/c and b/d as a simplified fraction. Test your program also with negative values. The function returns 1, if the operation succeeded, otherwise 0.

b) Write a function

```
int subFraction(int a, int c, int b, int d)
```

which prints the subtraction of a/c and b/d as a simplified fraction.

c) Write a function

```
int mulFraction(int a, int c, int b, int d)
```

which prints the multiplication of a/c and b/d as a simplified fraction.

The types uint16_t and uint32_t can be found in the library stdint.

a) Write a function that prints an unsigned 32-bit integer in binary form.

The display format should include a separator at byte boundaries.

Example:

print_bin(0) => 00000000.00000000.00000000.00000000

print_bin(1) => 00000000.00000000.00000000.00000001

print_bin(2) => 00000000.00000000.00000000.00000010

print_bin(43) => 00000000.00000000.00000000.00101011

print_bin(123456789) => 00000111.01011011.11001101.00010101

b) Write a function

```
uint16_t make_16bit(uint8_t least_significant, uint8_t
most_significant);
```

that combines the given bytes into a 16-bit integer and returns it as the result.

Note: In practice, the first argument contains 8 least significant bits of an unsigned 16-bit integer and the second contains the 8 most significant bits.

Example:

print_bin(1) => 00000000.00000000.00000000.00000001

print_bin(7) => 00000000.00000000.00000000.00000111

print_bin(make_16bit(7, 1)) => 00000000.00000000.00000001.00000111

Write a program to print the following pattern:

```
          C
        i   I
      s       s
    b           b
  e               e
s                   s
t s e b s i C i s b e s t
```

Write a program that, given a date, three ints (for example, 11 27 1997), will print the number of that day within its year: i.e. Jan 1st is always 1, Dec 31st is either 365 or 366.

The months of the year have lengths according to the following rules:

- The odd months up to and including month 7 have 31 days.
- The even months from 8 upwards, have 31 days.
- Month 2 has 28 days except in a leap year when it has 29 days.
- The rest of the months have 30 days.

Write a program to read a string from the console/stdin and count the number of chars, words, and lines, and print these quantities.

Write a program in C to add two numbers using pointers.

Test Data :
Input the first number : 5
Input the second number : 6
Expected Output :

```
The sum of the entered numbers is : 11
```

Write a program in C to print all permutations of a given string using pointers.

Expected Output :

```
The permutations of the string are :
abcd  abdc  acbd  acdb  adcb  adbc  bacd  badc  bcad  bcda
bdca  bdac  cbad  cbda  cabd  cadb  cdab  cdba  db
ca  dbac  dcba  dcab  dacb  dabc
```

Write a program in C to count the number of vowels and consonants in a string using a pointer.

Test Data :

Input a string: string

Expected Output :

```
Number of vowels : 1
Number of consonants : 5
```

Write a program in C to check whether two given strings are an anagram. You must use pointers in this task.

Test Data :
Input the first String : spare
Input the second String : pears
Expected Output :

```
 spare and pears are Anagram.
```

Write a C program to input any number from user and check whether n^{th} bit of the given number is set (1) or not (0).

Example

Input

```
Input number: 12
Input nth bit number: 2
```

Output

```
2 bit of 12 is set (1)
```

Write a C program to input any number from user and toggle or invert or flip n^{th} bit of the given number using bitwise operator.

Example

Input

```
Input number: 22
Input nth bit to toggle: 1
```

Output

```
After toggling nth bit: 20 (in decimal)
```

Write a C program to input any number from user and count number of trailing zeros in the given number using bitwise operator.

Example

Input

```
Input any number: 22
```

Output

```
Trailing zeros: 1
```

Write a C program to input a number from user and count total number of ones (1s) and zeros (0s) in the given number using bitwise operator.

Example

Input

```
Input any number: 22
```

Output

```
Output number of ones: 3
Output number of zeros: 29
```

Write a C program to input any number and check whether the given number is even or odd using bitwise operator.

Example

Input

```
Input number: 12
```

Output

```
12 is even
```

Write a C program to concatenate two strings in single string. You cannot use the built-in function here. You should use dynamic memory allocation at the heap.

Example

Input

```
Input string1: I love
Input string2: DIT632
```

Output

```
Concatenated string: I love DIT632
```

Write a C program to input any number from user and find highest order set bit of given number using bitwise operator.

Example

Input

```
Input any number: 22
```

Output

```
Highest order set bit in 22 is 4.
```

Write a C program to input marks of five subjects Physics, Chemistry, Biology, Mathematics and Computer, calculate percentage and grade according to given conditions:

If percentage >= 90% : Grade A
If percentage >= 80% : Grade B
If percentage >= 70% : Grade C
If percentage >= 60% : Grade D
If percentage >= 40% : Grade E
If percentage < 40% : Grade F

Example
Input

```
Input marks of five subjects: 95
```

```
95
97
98
90
```

Output

```
Percentage = 95.00
Grade A
```

The following program has two mistakes in it, please find them, explain why these are mistakes and correct them:

```c
#include <stdio.h>
#define MAX_SIZE 5 //Maximum size of the array
int main()
{
    int arr[MAX_SIZE]; //Declares an array size
    int i, num;
    //Enter size of array
    printf("Enter size of the array: ");
    scanf("%d", &num);
    //Reading elements of array
    printf("Enter elements in array: ");
    for(i=0; i<num; i++)
    {
        scanf("%d", &arr[i]);
    }
    printf("\nAll negative elements in array are: ");

    for(i=0; i<num; ++i)
    {
        //Printing negative elements
        if(arr[i] < 0)
        {
            printf("%d\t", arr[i]);
        }
    }
    return 0;
}
```

Tips! Execute the program and test it!

There are two mistakes in this program. Please find them, explain them and correct them:

```c
#include<stdio.h> //Used to include basic c library files
void main() //Used to execute the C application
{
```

```
    //declaring and defining the array variables
    int a[5] = {100,101,102,103,104};
    int b[5] = {105,106,107,108,109};

    //displaying the output
    printf("%d\n",a[100]);

    //and another element
    printf("%c\n",b[700]);
}
```

The program below contains two mistakes. Please find them, explain them and correct them.

```
#include<stdio.h> //Used to include basic c library files

void main()//main() method for executing the application
{
    //declaring and defining the variables
    int a=100;
    int b=200;

    //displaying the output
    printf("Sum of %d and %d is=%d\n",a,b,sum(a,b));//sum(a,b) is calling
method
}

//called method
int sum(int a, int b)
{
    return a*b; // making the calculation
}
```

The program below handles linked lists, but it is incomplete. I've forgotten to add the body of the function "length" and the code to delete the list. I marked the places with //TODO:. Please add the missing code and test it.

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

// structure describing one node of the list
struct node {
    int data;
    int key;

    struct node* next;
};
```

```
// pointing to the beginning of the list and the current node
struct node* head = NULL;
struct node* current = NULL;


int length() {
    // TODO: write the code to count the number of elements in the list

    return length;
}

//insert link at the first location
void insertFirst(int key, int data) {

    //create a link
    struct node* link = (struct node*)malloc(sizeof(struct node));
    link->key = key;
    link->data = data;

    if (isEmpty()) {
        head = link;
        head->next = head;
    }
    else {
        //point it to old first node
        link->next = head;

        //point first to new first node
        head = link;
    }
}

//delete first item
struct node* deleteFirst() {

    //save reference to first link
    struct node* tempLink = head;

    if (head->next == head) {
        head = NULL;
        return tempLink;
    }

    //mark next to first link as first
    head = head->next;

    //return the deleted link
    return tempLink;
```

```
}

//display the list
void printList() {

    struct node* ptr = head;
    printf("\n[ ");

    //start from the beginning
    if (head != NULL) {

        while (ptr->next != ptr) {
            printf("(%d,%d) ", ptr->key, ptr->data);
            ptr = ptr->next;
        }
    }

    printf(" ]");
}

void main() {
    insertFirst(1, 10);
    insertFirst(2, 20);
    insertFirst(3, 30);
    insertFirst(4, 1);
    insertFirst(5, 40);
    insertFirst(6, 56);

    printf("Original List: ");

    //print list
    printList();

    // TODO: write the code to delete the entire list

    printf("\nList after deleting all items: ");
    printList();
}
```

TinkerCad/Arduino questions

Create a system with the proximity/distance sensor. You should use the sensor to get the information about the distance to an obstacle. You should use four LEDs to indicate how close the obstacle is from the sensor.

If the object is out of range, then no LEDs should be turn on. If the object is in range, you should turn on 1, 2 or 3 LEDs, depending on the distance. If the object is too close, you should turn on four LEDs.

You should deliver the following:

- Picture of the board and the wiring (correct answer: 2 points)
- Code, as a document or .c file
    - Where you define the distance when you turn on and off the LEDs (1 point)
    - Where you define the function that receives the value of the sensor (3 points)
    - Comments (1 point)

Create a system with the force sensor and a buzzer. The system should use the buzzer to make noise when a force is applied to the force sensor. The frequency of buzzing should be based on the force applied to the sensor, in the following way:

Force	Buzzer frequency
0	0 – buzzer off
0.5 N	1 sek
1 N	0.3 sek
2 N	0.2 sek
5 N	0.1 sek
> 6 N	0.05 sek

You can use interrupts or a loop for this exercise.

You should deliver the following:

- Picture of the board with the wiring (2 points)
- Code, as a document or .c file
    - Where you use #define or constants to predefine the buzzer frequency and the force (2 points)
    - Where you have a function that reads the force and sets the buzzer frequency (2 points)
    - Where you describe why your solution guarantees that the buzzer will buzz with the desired frequency (2 points). This should be described as a comment in the source code.
    - Comments (1 point)
