RECITATION 11

Q1. Write a program with following screen output:

```
Current Month     | Previous Month
------------------------------
January           | December
Februari          | January
March             | February
April             | March
May               | April
June              | May
July              | June
August            | July
September         | August
October           | September
November          | October
December          | November
```

Make use of:

- an enum type “month”:

  enum t_month {Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec};

- a function “lastMonth” that has the current month (type enum t_month) as input and returns the previous month,
- a function “printMonth” that takes a variable of the type enum t_month as input and prints that month to the screen,
- a for loop to run through all 12 months.

Q2. Write a program “next-day” with:

- a function “read” that asks the user to enter a day and a month as integers and returns those values to the main function using pointers,
- a function “nextDay” that has the current day and month as arguments, calculates the next day and returns that next day to the main function (assume the day is not part of leap year),
- a function “printDay” that takes a day and a month as input and prints them to the screen,
- use an enum type for the months.

```
Enter the current day and month (as integers): 29 3
The current day is: March 29
The next day is: March 30
```

```
Enter the current day and month (as integers): 30 4
The current day is: April 30
The next day is: May 1
```

Q3. Write a program that reads an integer, stores it in a variable of the type short int and prints the integer as a sequence of 4 nibbles (nibble = 4 bits). The nibbles can be printed in decimal or in hex format. Make sure the leftmost nibble is printed first and the rightmost is printed last. Determining the values of the nibbles needs to be done using bit operations only. Make use of functions in your program. For example;

- input: 23043 (=0x5A03) output: 5 10 0 3
- input: -21345 (=0xAC9F) output: 10 12 9 15

Q4. Write a function with return value that rotates the bits of a short int 4 positions to the left. The bits that are shifted out at the left side, must be reentered at the right side. Rotating needs to be done inside the variable that was entered. Write a program that uses this function. For example;

- input: 20480 output: 5 (20480 = 0x5000 -> rotation yields: 0x0005)
- input: 23043 output: -24523 (23043 = 0x5A03 -> rotation yields: 0xA035)
- input: -24523 output: 858 (-24523 = 0xA035 -> rotation yields: 0x035A)

Q5. Write a function that prints the binary representation of a short int using bit operations. Write a program that uses this function.

- input: 23043 output: 0101 1010 0000 0011
- input: -24523 output: 1010 0000 0011 0101

Q6. Consider a register of the CANSPI MCP2515 chip. The register contains 8 bits out of which the 2 MSB's (bits 6 and 7) are the "Synchronization Jump Width Length Bits" and the 6 LSB's (bits 0 till 5) are the "Baud Rate Prescaler Bits" (see figure below).

REGISTER 5-1: CNF1 – CONFIGURATION 1 (ADDRESS: 2Ah)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **SJW<1:0>:** Synchronization Jump Width Length bits
11 = Length = 4 x TQ
10 = Length = 3 x TQ
01 = Length = 2 x TQ
00 = Length = 1 x TQ

bit 5-0 **BRP<5:0>:** Baud Rate Prescaler bits
TQ = 2 x (BRP + 1)/Fosc

Write a program that asks the user to enter a number that fits this register (1 byte) and that prints the corresponding “Baud Rate Prescaler Bits” (hex and binary format) and the “Synchronization Jump Width Bits” (hex and binary format).

Q7. Write a function that produces a (pseudo) random number in the interval [1, 32767]. Make use of the following algorithm:

- Use a SEED different from 0 (for example, 3254).
- Store this seed number into a 16 bit static short int variable (static is needed to continue with the changed value when the function is called several times).
- Perform a XOR operation on the bits 14 and 13 (the bits are numbered from left to right as follows: 15, 14, 13, …, 2, 1, 0).
- Shift the number 1 bit to the left and fill the rightmost bit with the result of the XOR operation.
- Make sure bit 15 remains 0 (to keep the number in the correct range).
- The number you have now is a pseudo random number.

Write a program that calls this function 10 times. Check the result.