Lab 02 Report

Muhammad Hassan Shahid: ms10290

Muhammad Saad Malik: mm10357

Task 0

```c
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
void myPrintf (const char *fmt , ...){
        char buffer[1024];
        va_list args;
        va_start (&: args, fmt);
        int len = vsnprintf (buffer, sizeof(buffer), fmt, args);
        va_end (&: args);
        HAL_UART_Transmit(huart: &huart2, pData: (uint8_t*)buffer, Size: len, Timeout: HAL_MAX_DELAY);
    }
/* USER CODE END 0 */
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   STM32CUBE RTOS

Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, W
ld
```

Task 01

```c
while (1)
{
  /* USER CODE END WHILE */
  HAL_UART_Transmit(huart: &huart2, pData: (uint8_t*)"Hello, World!\r\n", Size: 15, Timeout: HAL_MAX_DELAY);
  /* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
}
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   STM32CUBE RTOS

Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Value of x = 42, y = 3.14
Va
```

Task 02

```c
/* USER CODE BEGIN 2 */
int a=2;
int b=9;
int lhs= (a+b)*(a+b);
int rhs= a*a + 2*a*b + b*b;

/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE BEGIN 3 */
  if (lhs == rhs){
    myPrintf(fmt: "Algebra works!\r\n");
  } else {
    myPrintf(fmt: "Algebra fails!\r\n");
  }
  myPrintf (fmt: " Value of LHS = %d, RHS = %d\r\n", lhs, rhs);
  /* USER CODE END WHILE */
}
/* USER CODE END 3 */
```

```
 Value of LHS = 121, RHS = 121
Algebra works!
 Value of LHS = 121, RHS = 121
Algebra works!
 Value of LHS = 121, RHS = 121
Algebra works!
 Value of LHS = 121, RHS = 121
Algebra works!
 Value of LHS = 121, RHS = 121
Algebra works!
 Value of LHS = 121, RHS = 121
Algebra works!
 Value
```

Task 03

```c
/* USER CODE BEGIN 2 */
char str[] = "Microcontrollers";
int key = 1029010357;


/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE BEGIN 3 */
  for (int i = 0; str[i] != '\0'; i++) {
    str[i] = str[i] + (key % 256);
  }
  myPrintf (fmt: " Encrypted = %s\r\n", str, key);
  for (int i = 0; str[i] != '\0'; i++) {
    str[i] = str[i] - (key % 256);
  }
  myPrintf (fmt: " Decrypted = %s\r\n", str, key);

  /* USER CODE END WHILE */
}
```

```
Decrypted = Microcontrollers
Encrypted = '$$#)'$!!'(
Decrypted = Microcontrollers
Encrypted = '$$#)'$!!'(
Decrypted = Microcontrollers
Encrypted = '$$#)'$!!'(
Decrypted = Microcontrollers
Encrypted = '$$#)'$!!'(
Decrypted = Microcontrollers
Encrypted = '$$#)'$!!'(
Decrypted = Microcontrollers
Encrypted = '$$#)'$!!'(
Decrypted = Microco
```

Task 04

```
/* USER CODE BEGIN 2 */
 int A[2][2] = {
      [0]={[0]=1, [1]=2},
      [1]={[0]=3, [1]=4}
  };

  int B[2][2] = {
      [0]={[0]=5, [1]=6},
      [1]={[0]=7, [1]=8}
  };

  int C[2][2] = {0};

  for (int i = 0; i < 2; i++)
  {
      for (int j = 0; j < 2; j++)
      {
          C[i][j] = 0;
          for (int k = 0; k < 2; k++)
          {
              C[i][j] += A[i][k] * B[k][j];
          }
      }
  }
```

```
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE BEGIN 3 */
  myPrintf(fmt: "Matrix A:\r\n");
  for (int i = 0; i < 2; i++)
  {
      for (int j = 0; j < 2; j++)
      {
          myPrintf(fmt: "%d ", A[i][j]);
      }
      myPrintf(fmt: "\r\n");
  }

  myPrintf(fmt: "\r\nMatrix B:\r\n");
  for (int i = 0; i < 2; i++)
  {
      for (int j = 0; j < 2; j++)
      {
          myPrintf(fmt: "%d ", B[i][j]);
      }
      myPrintf(fmt: "\r\n");
  }

  myPrintf(fmt: "\r\nMatrix C (A x B):\r\n");
  for (int i = 0; i < 2; i++)
  {
      for (int j = 0; j < 2; j++)
      {
          myPrintf(fmt: "%d ", C[i][j]);
      }
      myPrintf(fmt: "\r\n");
  }

  /* USER CODE END WHILE */
}
/* USER CODE END 3 */
```

```
Matrix A:
1 2
3 4

Matrix B:
5 6
7 8

Matrix C (A × B):
19 22
43 50
```

Task 05

```
//task 5
 int num;
```

```
// task 5
myPrintf(fmt: "3-digit Armstrong numbers are:\r\n");
for (num = 100; num <= 999; num++) {
    int hundreds = num / 100;         // X
    int tens = (num / 10) % 10;       // Y
    int units = num % 10;             // Z

    // (b) Compute sum of cubes
    int sum_cubes = hundreds*hundreds*hundreds
                  + tens*tens*tens
                  + units*units*units;

    // Compare to original number
    if (sum_cubes == num) {
        // (c) Print Armstrong number
        myPrintf(fmt: "%d\r\n", num);
    }
}
```

```
3-digit Armstrong numbers are:
153
370
371
407
```

Evaluation Questions:

1. Role of USART2

USART2 on the STM32F3 Discovery board is a peripheral used for **UART communication** (asynchronous serial communication). It provides the interface for sending and receiving serial data between the STM32 microcontroller and external devices such as a PC or USB-UART module.

UART Mode: Allows asynchronous transmission and reception without a clock signal.

Pins:

PA2 → USART2_TX (Transmit)

PA3 → USART2_RX (Receive)

Function:

TX (Transmit): Sends data from the STM32 to the external device.

RX (Receive): Receives data from the external device into the STM32.

Use Case: Commonly used for debugging, logging, or command/control communication via a USB-UART interface.

2. Why TX and RX Lines Must Be Crossed

When connecting the STM32F3 board to another UART device (like a USB-UART module), the transmit and receive lines must be crossed to ensure proper communication:

STM32 Pin	Function	USB-UART Pin
TX	Transmit	RX
RX	Receive	TX

Reason: Each device's **transmit line must connect to the other device's receive line**.

If TX and RX are not crossed, both devices could be transmitting on the same line or trying to receive from an inactive line, causing communication failure.

Connection Summary:

```
STM32_TX ↔ USB-UART_RX
STM32_RX ↔ USB-UART_TX
```

This ensures proper two-way UART communication between the STM32F3 and the USB-UART module.