Задача 1. Предположим, что процессор выполняет вычисления над 8-битными двоичными числами в дополнительном коде. Какие значения примут флаги Z, N, C, V в результате выполнения операций (все числа записаны в десятичной системе счисления).

- 49 + 79
- 211 + 45

Ответ.

```
Z=0 N=1 C=0 V=1
Z=1 N=0 C=1 V=0
```

Задача 2. Чем отличается микропроцессор от микроконтроллера (запишите ответ одним предложением)?

Ответ. Микроконтроллер кроме микропроцессора содержит энергонезависимую память для хранения программы и данных (Flash или EEPROM), оперативную память, коммуникационные интерфейсы (UART, SPI, USB и пр.), таймеры, прочие периферийные устройства.

Микроконтроллер может быть как гарвардской, так и фон-неймановской архитектуры.

Задача 3. На языке ассемблера ARMv7 напишите функцию

```
unsigned satsum(unsigned v1, unsigned v2);
```

которая выполняет сложение с насыщением двух беззнаковых чисел. То есть, если сложение вызывает переполнение, результатом будет максимальное число данного типа.

Ответ.

```
        .align  4
        .text
        .global satsum
satsum:
        stmfd   sp!, {fp, lr}
        adds    r0, r0, r1
        movcs   r0, #-1
        ldmfd   sp!, {fp, pc}
```

Задача 4. Для чего необходим позиционно-независимый код (запишите ответ одним предложением)?

Ответ. Позиционно-независимый код используется в разделяемых библиотеках (например, стандартная библиотека языка Си libc.so) и позволяет не модифицировать сегмент кода библиотеки (.text) при загрузке на разные базовые адреса в разных процессах, что позволяет использовать одну и ту же физическую память для всех копий библиотеки, загруженных во все процессы, что очень существенно экономит физическую оперативную память.

Задача 5. Дано описание:

```
struct A {
    unsigned short f1;
    long long f2;
    char f3[7];
};
struct B {
    struct A a1;
    short a2[3];
    struct A a3;
    char a4;
};
```

Чему равен `sizeof(struct B)` на платформе ARM при стандартных настройках выравнивания? Каков минимальный `sizeof(struct B)` можно получить при изменении порядка полей в структурах? Ответ запишите в виде двух чисел.

Ответ. Размер структур: `sizeof(struct A) == 24`, `sizeof(struct B) == 64`. После размещения полей структур в порядке уменьшения требований к выравниванию получим `sizeof(struct A) == 24`, `sizeof(struct B) == 56`.

Задача 6. Почему в архитектуре ARM нельзя загрузить в регистр произвольное константное значение инструкцией

```
        mov     r0, #VALUE
```

Ответ. Любая инструкция кодируется 32-битами, поэтому под представление значения `VALUE` отводится меньше 32 бит, так как требуется еще кодировать код операции и номер регистра.

Задача 7. Запишите инструкцию `stmfd sp!, {r0, r1}` в виде двух инструкций `str`.

Ответ.

```
str r1, [sp, #-4]!
str r0, [sp, #-4]!
```

Задача 8. На языке ассемблера ARM напишите функцию `process`, которая вычисляет выражение

```
    C = (A + B) << 1;
```

где `A`, `B`, `C` — глобальные внешние переменные типа `long long`. Глобальные переменные находятся в секции `.data`, код функции `process` должен находиться в секции `.text`.

Ответ.

```
        .text
        .align  4
        .global process
process:
        stmfd   sp!, {fp, lr}
        ldr     r2, Aaddr
        ldmia   r2, { r0, r1 }
        ldr     r2, Baddr
        ldmia   r2, { r2 ,r3 }
        adds    r0, r0, r2
        adc     r1, r1, r3
        movs    r0, r0, lsl #1
        adc     r1, r1, r1
        ldr     r2, Caddr
        stmia   r2, { r0, r1 }
        ldmfd   sp!, {fp, pc}
Aaddr:  .word   A
Baddr:  .word   B
Caddr:  .word   C
```

Задача 9. На языке ассемблера ARM напишите функцию `reverse` с прототипом

```
void reverse(int *data, int count);
```

Функция меняет порядок следования элементов массива `data` на противоположный (6 инструкций в цикле).

Ответ.

```
        .align  4
        .text
        .global reverse
```

```
reverse:
        stmfd   sp!, { fp, lr }
        add     r2, r0, r1, lsl #2
        cmp     r1, #1
        ble     done
loop:
        ldr     r1, [r0]
        ldr     r3, [r2, #-4]
        str     r1, [r2, #-4]!
        str     r3, [r0], #4
        cmp     r0, r2
        blo     loop
done:
        ldmfd   sp!, { fp, pc }
```

Задача 10. На языке ассемблера ARM напишите функцию `myatoi` с прототипом

```
int myatoi(const char *str);
```

Функция преобразовывает строку, содержащую число в десятичной записи, в целое число. Строка содержит только цифры. Переполнение игнорировать. Умножение реализовывать с помощью сложения и сдвигов (7 инструкций в цикле).

Ответ.

```
        .text
        .align  4
        .global myatoi
myatoi:
        stmfd   sp!, { fp, lr }
        mov     r1, r0
        eor     r0, r0, r0
        ldrb    r2, [r1], #1
        tst     r2, r2
        beq     done
loop:
        add     r0, r0, r0, lsl #2
        add     r0, r0, r0
        sub     r2, r2, #'0'
        add     r0, r0, r2
        ldrb    r2, [r1], #1
        tst     r2, r2
        bne     loop
done:
        ldmfd   sp!, { fp, pc }
```