

## Семинар 5 - Ассемблер AArch64.

### Задача

Реализуйте функцию `_print_list` на ассемблере AArch64. Функция принимает указатель на первый элемент связного списка и должна вывести в консоль содержимое полей `element` всех элементов используя функцию `puts`

### Заголовочный файл:

```
1  typedef struct list_entry {
2      struct list_entry* next;
3      const char* element;
4  } list_entry;
5
6  void print_list(list_entry* first);
```

### Заготовка:

```
1  _print_list:
2      sub sp, sp, 32
3      stp x29, x30, [sp, +16]
4      mov x29, sp
5      # <ваш код>
6
7
8
9      # Заготовка: печать первого элемента
10     ldr x0, [x0, 8] # x0 = first->element
11     bl _puts      # puts(first->element)
12
13
14
15
16     # </ваш код>
17     ldp x29, x30, [sp, +16]
18     add sp, sp, 32
19     ret
```

## Инструкции обработки данных

**ADD** <Rd>, <Rn>, <Rm>:  $Rd = Rn + Rm$ ;  
**SUB** <Rd>, <Rn>, <Rm>:  $Rd = Rn - Rm$ ;  
**MUL** <Rd>, <Rn>, <Rm>:  $Rd = Rn * Rm$ ;

## Инструкции load / store

**LDR** <Rt>, [<Rn>]:  $Rt = *Rn$ ;  
**LDR** <Rt>, [<Rn>, <imm>]:  $Rt = *(Rn + imm)$ ;  
**LDR** <Rt>, [<Rn>, <imm>]!:  $Rt = *(Rn += imm)$ ;  
**LDR** <Rt>, [<Rn>], <imm>:  $Rt = *(Rn)$ ;  $Rn += imm$ ;  
**STR** <Rt>, [<Rn>]:  $*Rn = Rt$ ;  
**STR** <Rt>, [<Rn>, <imm>]:  $*(Rn + imm) = Rt$ ;  
**STR** <Rt>, [<Rn>, <imm>]!:  $*(Rn += imm) = Rt$ ;  
**STR** <Rt>, [<Rn>], <imm>:  $*(Rn) = Rt$ ;  $Rn += imm$ ;  
**STP** <Ra>, <Rb> -||-: Аналогично, но сохраняет два регистра  
**LDP** <Ra>, <Rb> -||-: Аналогично, но загружает два регистра

## Инструкции перехода

**B** <label>: **goto** label  
**BL** <label>: **call**, или **goto** label + r30 = <return address>  
**RET**: **return**, или **goto** r30

## Логические инструкции

**AND** <Rd>, <Rn>, <Rm>:  $Rd = Rn \& Rm$ ;  
**ORR** <Rd>, <Rn>, <Rm>:  $Rd = Rn | Rm$ ;  
**EOR** <Rd>, <Rn>, <Rm>:  $Rd = Rn \wedge Rm$ ;

## Условные суффиксы

Чтобы изменить флаги, добавь суффикс **S**

<b>.EQ</b> : Если равно	<b>.NE</b> : Если не равно
<b>.HS</b> : Если беззнаково $\geq$	<b>.LO</b> : Если беззнаково $<$
<b>.MI</b> : Если отрицательно	<b>.PL</b> : Если положительно или ноль
<b>.VS</b> : Если переполнение	<b>.VC</b> : Если нет переполнения
<b>.HI</b> : Если беззнаково $>$	<b>.LS</b> : Если беззнаково $\leq$
<b>.GE</b> : Если знаково $\geq$	<b>.LT</b> : Если знаково $<$
<b>.GT</b> : Если знаково $>$	<b>.LE</b> : Если знаково $\leq$