Xavier Kuehn
Homework 3
COEN 20

1. Translate each of the following into equivalent C assignment statements and variable declarations.

A)
```
uint32_t green;
green += 1;
uint64_t violet;
violet = green;
```

B)
```
uint16_t red;
uint32_t blue[];
uint32_t purple;
*(blue + sizeof(purple) * 2) = red;
```

C)
```
uint32_t black;
int32_t white;
black += (white * 4);
black = 0;
```

2. Given the C declaration statements, translate each into assembly language.
(a → c)

LDR R0,=0
LDRB R1,=u8
LDRB R1,[R1]
STRB R0,[R1]
	LDR R0,=0
	LDRH R1,=u16
	LDRH R1,[R1]
	STRH R0,[R1]
		LDR R0,=0
		LDR R1,=u32
		LDR R1,[R1]
		STR R0,[R1]

LDR R2,=0
LDRD R0,R1,[R2]
LDRD R2,=u64
LDRD R2,R3,[R2]
STRD R0,R1,[R2]
	LDR R0,=u8
	LDRB R0,[R0]
	LDR R1,=u16
	LDRH R1,[R1]
	LDRH R1,[R0]
		LDR R0,=u8
		LDRB R0,[R0]
		LDR R1,=u32
		LDR R1,[R1]
		STR R0,[R1]

LDR R0,=u32
LDR R0,[R0]
LDR R1,=u16
LDRH R1,[R1]
STR R1,[R0]
	LDR R2,=u64
	LDRD R0,R1,[R2]
	LDR R2,=u32
	LDR R1,[R1]
	STR R0,R1,[R2]
		LDR R0,=u8
		LDRB R0,[R0]
		LDR R1,=u32
		LDR R1,[R1]
		STRB R1,[R0]

LDR R0,=u8 LDRB R0,[R0]	LDR R0,=u8 LDRB R0,[R0]	LDR R0,=u16 LDRH R0,[R0]

`LDR  R1,=u16`
`LDRH R1,[R1]`
`STRB R1,[R0]`
	`LDR  R1,=u32`
	`LDR  R1,[R1]`
	`STRB R1,[R0]`
		`LDR  R1,=u32`
		`LDR  R1,[R1]`
		`STRH R1,[R0]`

4. Given the C declaration statements, translate each assignment statement into assembly language.

a)
```
LDR  R0,=s32
LDR  R1,=ps32
LDR  R1,[R1]
STR  R0,[R1]
```

b)
```
LDR  R0,=s8
LDR  R1,=ps8
LDRB R1,[R1]
STRB R0,[R1]
```

c)
```
LDR  R0,=ps8
ADD  R0,R0,4
STRB R0,[R0]
```

d)
```
LDR  R0,=ps32
ADD  R0,R0,4
STR  R0,[R0]
```

e)
```
LDR  R0,=ps32
LDR  R1,=0
STR  R1,[R0]
```

f)
```
LDR  R0,=ps8
LDR  R1,=0
STR  R1,[R0,4]
```

g)
```
LDR  R0,=ps32
LDR  R1,=0
STR  R1,[R0,4]
```

h)
```
LDR  R0,=ps8
LDR  R1,=ps32
LDR  R2,=0
STR  R2,[R0,R1]
```

```
   i) LDR  R0,=s32
      LDRB R1,[R0]
      LDR  R0,=0
      STRB R0,[R1,4]
```

7. Write an assembly function to swap the contents of its two actual arguments.
Function prototype: `void Swap32(int32_t *p1, int32_t *p2);`

```
int32_t tmp;    // R2
tmp = *p1;      // LDR    R2,[R0] ; reg ← mem
*p1 = *p2;      // STR    R1,[R0] ; reg → mem
*p2 = tmp;      // STR    R2,[R1] ; reg → mem

Swap32:   LDR  R2,[R0]
          STR  R1,[R0]
          STR  R2,[R1]
          BX   LR
```