1.

a.: LDR R5, [R7, #4]

ADD R5, R5, R2

ADD R5, R5, R1

ADD R0, R0, R5

LDR R5, [R7, #4]

LDR R5, [R6, R5]

SUB R0, R1, R5

b. **int** a[12];

a[0]=2;

a[4]=4;

a[8]=6;

a[12]=1;

**int** b[12];

**int** min=a[0];

**int** index=0;

**while**(index<=11){

**for**(**int** i=0;i<12;i++){

**if**(min>=a[i]){

min=a[i];

}

}

b[index]=min;

index++;

}

a=b;

**return** a;

e:

1. R3=0x55555555, r3=0101 0101 0101 0101 0101 0101 0101 0101

R4=0x12345678, r4=0001 0010 0011 0100 0101 0110 0111 1000

1. R3=0xBEADFEED, r3=1011 1110 1010 1101 1111 1110 1110 1101

R4=0xDEADFADE, r4=1101 1110 1010 1101 1111 1010 1101 1110

2.13.1: OR r5, r4, r3, LSL#4

A: r3,LSL #4 => 0101 0101 0101 0101 0101 0101 0101 0000

r4=0001 0010 0011 0100 0101 0110 0111 1000

So the results for OR r5,r4, r3, LSL#4 should be

0101 0111 0111 0101 0101 0111 0111 1000

B: r3,LSL #4 => 1110 1010 1101 1111 1110 1110 1101 0000

r4= 1101 1110 1010 1101 1111 1010 1101 1110

1111 1110 1111 1111 1111 1110 1101 1110

2.13.2: MVN r3, #1 => r3=1111 1111 1111 1111 1111 1111 1111 1111

a. R4, LSL #4 => 0010 0011 0100 0101 0110 0111 1000 0000

AND r5, r3, r4, LSL #4

=>r5=0000 0001 0100 0101 0100 0101 0000 0000

1. R4, LSL #4 => 1110 1010 1101 1111 1010 1101 1110 0000

AND r5, r3, r4, LSL #4

=>r5=1010 1010 1000 1101 1010 1100 1110 0000

2.13.3: MOV R5, 0XFFEF => R5=1111 1111 1110 1111

1. A. R3, LSR #3 =>R3= 0000 1010 1010 1010 1010 1010 1010 1010

AND R5, R5, R3, LSR #3

2.13.4: R0=0X0000A5A5 =>R0= 0000 0000 0000 0000 1010 0101 1010 0101

R1=0X00005A5A =>R1= 0000 0000 0000 0000 0101 1010 0101 1010

1. R0, LSL #1 => 0000 0000 0000 0001 0100 1011 0100 1010

ORR R2, R1, R0, LSL #1 =>

0000 0000 0000 0001 0101 1011 0101 1010

1. R0, LSR #1 => 0000 0000 0000 0000 0101 0010 1101 0010

AND R2, R1, R0, LSR #1 =>

0000 0000 0000 0000 0101 0010 0101 0010

2.13.5: R0= 0XA5A50000 => RO= 1010 0101 1010 0101 0000 0000 0000 0000

R1= 0XA5A50000 => R1= 1010 0101 1010 0101 0000 0000 0000 0000

1. R0, LSL #1 => 0100 1011 0100 1010 0000 0000 0000 0000

ORR R2, R1, R0, LSL #1 =>

1110 1111 1110 1111 0000 0000 0000 0000

1. R0, LSR #1 => 0101 0010 1101 0010 1000 0000 0000 0000

AND R2, R1, R0, LSR #1 =>

0000 0000 1000 0000 0000 0000 0000 0000

2.13.6: R0= 0XA5A5FFFF => R0= 1010 0101 1010 0101 1111 1111 1111 1111

R1= 0XA5A5FFFF => R1= 1010 0101 1010 0101 1111 1111 1111 1111

1. R0, LSL #1 => 0100 1011 0100 1011 1111 1111 1111 1110

ORR R2, R1, R0, LSL #1 =>

1110 1111 1110 1111 1111 1111 1111 1111

1. R0, LSR #1 => 0101 0010 1101 0010 1111 1111 1111 1111

AND R2, R1, R0, LSR #1 =>

0000 0000 1000 0000 1111 1111 1111 1111

f:

2.16.1: r2=0

2.16.2: r2=0

2.16.3:

2.16.4: r2=0

2.16.5: r2=0

G:

2.17.4:

a. r2=2, r1=9,

r2=4, r1=8,

……

R2=20, r1=0,

Done

So r2=20 after the loop a

1. It’s not going to jump out of the loop

H:

2.18.2:

a.

LOOP:CMP R2, #10

BLT B1

HALT

B1: ADD R0, R0, R1

ADD R2, R2, #1

B LOOP

b. B2: CMP r0, #10

BLT B1

HALT

B1: ADD r12, r3, r0, LSL #2

ADD R4, R0, R1

STR R4, [R12, #0]

ADD R0, R0, #1

B B2