Homework #2

Due at Thursday midnight, April 1, 2019

1. Practice Problem

2.23 fun1에서 shift는 비부호형 변수 word에 실행되고, 따라서 논리연산이다. fun2에서 shift는 word를 int로 캐스팅한 후에 실행되어 산술적 연산이 된다.

A:

w fun1(w) fun2(w)

0x00000076 0x00000076 0x00000076

0x87654321 0x00000021 0x00000021

0x000000C9 0x000000C9 0xFFFFFFC9

0xEDCBA987 0x00000087 0xFFFFFF87

B: 함수 fun1은 인자의 하위 8비트에서 값을 추출해서 0에서 255 사이의 정수를 만든다. 함수 fun2는 인자의 하위 8비트에서 값을 추출하지만, 이 경우에는 부호 확장을 실행한다. 결과 값은 -128에서 127사이의 숫자가 될 것이다.

2.30

Determine whether arguments can be added without overflow

Int tadd\_ok(int w,int y){

Int sum=x+y;

Int neg\_over=x<0 && y<0 && sum>=0;

Int pos\_over=x>=0 && y>=0 && sum<0;

Return !neg\_over && !pos\_over;

}

2.31

식 (x+y)-x가 오버플로우의 발생여부와 상관없이 y가 되고 (x+y)-y는 x가 되기 떄문이다.

2.36

Int tmult\_ok(int x,int y){

Int 64\_t pll=(int64\_t) x\*y;

Return pll ==(int) pll;

}

2.37

A. 도움이 되지 않는다. Malloc을 사용하면 32비트 비부호형 값으로 변환되기 때문이다.

B. malloc함수를 사용하면 2^32바이트 이상을 갖는 블록을 할당할 수 없으며, 따라서 이 크기의 메모리를 복사하거나 할당하려는 시도를 할 이유가 없다. 대신 함수는 동작을 멈추고 NULL을 리턴해야한다.

Unit64\_t required\_size = ele\_cnt\*(uint64\_t) ele\_size;

Size\_t request\_size = (size\_t) required\_size;

If(required\_size != request\_size)

Return NULL;

Void \* result = malloc(request\_size);

If(result==NULL)

Return NULL;

1. Answer to following equation to YES or NO. Justify your answers.
2. int x = foo();
3. int y = bar();
4. unsigned ux = (unsigned) x;
5. unsigned uy = (unsigned) y

* 𝒙<𝟎⇒(𝒙∙𝟐)<𝟎 No 만약 4비트 1000의 두배를 한다면 10000이 된다 (x\*2)=0이되므로 성립이 안된다.
* 𝒖𝒙≥𝟎 Yes 비부호형 수이므로 항상 0이상이다
* (𝒙 & 𝟕)[[1]](#footnote-1)==𝟕⇒(𝒙≪𝟑𝟎)<𝟎 Yes x&7==7이므로 맨마지막 3자리의 숫자가 111이고 32비트이므로 30칸 쉬프트를 하면 msb 1이므로 음수이다.
* 𝒖𝒙≻−𝟏 Yes
* 𝒙>𝒚⇒−𝒙<−𝒚 No y가 Tmin일경우 -를 취해도 그 값이 같기 때문에 -x<-y가 된다.
* 𝒙∙𝒙≥𝟎 No x가 0xFFFF일경우 곱하면 0xFFFE0001이되므로 음수가 된다.
* 𝒙>𝟎 && 𝒚>𝟎⇒𝒙+𝒚>𝟎 No x=4000 0000 y=4000 0000이되므로 x+y=8000 0000즉 msb가 1이므로 음수가된다.
* 𝒙≥𝟎⇒−𝒙≤𝟎 yes
* 𝒙≤𝟎⇒−𝒙≥𝟎 no Tmin일경우 -를 취해도 -x<=0이다.
* (𝒙 |−𝒙)≫𝟑𝟏==−𝟏 yes x|-x 를 할경우 msb가 무조건 1이되기떄문에 shift를 하면 1111…1이되므로 -1이다.
* 𝒖𝒙≫𝟑==𝒖𝒙∕𝟖 yes unsigned이면 무조건 같다
* 𝒙≫𝟑==𝒙∕𝟖 No Shift를 하면 x가 음수일 때 버림을 하고, 나눗셈을 하면 0에 가까워지도록 올림이나 내림을 한다. 따라서 x가 음수일 때 달라진다.
* 𝒙 & (𝒙−𝟏) !=𝟎 NO 1000일경우 x-1은 0111이므로 &는 0이된다.

1. In the following questions assume the variables **a** and **b** are signed integers and that the machine uses two’s complement representation. Also assume the MAX\_INT is the maximum integer, MIN\_INT is the minimum integer, and W is one less than the word length (e.g., W = 31 for 32-bit integers).

Match each of the descriptions on the left with a line of code on the right (write in the number). Justify your answer.

* 1. One’s complement of a (5) MIN\_INT + MAX\_INT은 1111…1이므로 a^을하면 1의 보수를 취하는것과 같다.
  2. a (1) x^y=(x&~y)|(~x&y)이므로 x=(a^b) y=b 즉 a^b^b=a^0=a
  3. a & b (9) ~(~a | (b ^ (MIN\_INT | MAX\_INT))) == a & ~(b^111……. 111) ~(b^ 111…111) == b 따라서 9번은 a&b
  4. a \* 7 (2) a<<3이 a\*8이고 ~a+1=-a이므로 d와 같다.
  5. a / 4 (4) 책에 의하면 (x<0 ? x+(1<<k)-1:x)>>k 여기서 k=2를하면 (4)과 같고 이것은 (e)와 같다.
  6. (a < 0) ? 1 : -1 (7)

If(a<0) a>>W이 1111..1

<<1을하면 111…0 ~을 하면 1이된다

If(a>0) a>>W이 00000..0

<<1을 해도 똑같으니 ~을 하면 -1이된다.

1. ((a^b) & ~b)|(~(a^b) & b)
2. 1+(a<<3) + ~a
3. (a<<4)+(a<<2)+(a<<1)
4. ((a<0)?(a+3): a)>>2
5. a^(MIN\_INT + MAX\_INT)
6. ~((a|(~a+1))>>W)& 1
7. ~((a>>W)<<1)
8. a>>2
9. ~(~a | (b ^ (MIN\_INT | MAX\_INT)))

(6)~a+1=-a (a|-a)의 msb는 무조건1

>>W을 해주면 111..1이되므로 ~을 취하면 0이므로 &1을할시 무조건 0

# Deliverables

Work your report with a word processor at your preference. Name you file to yourSchoolId.[doc|hwp] and upload your report to iCampus. Please do not zip your file or your will get no points for the homework.

1. ‘==’ has a higher precedence than ‘&’ [↑](#footnote-ref-1)