Student: **Collegekaartnummer:**

Tentamen Computersystemen

baiCOSY06 2e jaar bachelor AI, 2e semester 27 september 2011

vraag 1

In deze vraag wordt er gevraagd om equivalente expressies met logische operatoren bij elkaar te zoeken. Er mag worden aangenomen dat de twee variabelen variables `a` en `b` *signed integers* zijn. Verder mag men er van uitgaan de machine de standaard *two's complement* representatie gebruikt om gehele getallen te representeren. .De constante `INT_MAX` bevat de waarde van de *maximum integer*, `INT_MIN` de waarde van de *minimum integer*, en `W` is de woord lengte minus één (bijvoorbeeld `W` = 31 voor 32-bit *integers*). Hieronder staan drie kolommen. Vind voor iedere expressie in de linker column de corresponderende expressie in de rechter column (het antwoord bestaat dus enkel uit een letter), en zet die in de middelste column.

	Correspondeert met:	
1. `a`		a. `~(~a | (b ^ (INT_MIN + INT_MAX)))`
2. `a & b`		b. `((a ^ b) & ~b) | (~(a ^ b) & b)`
4. a * 7		c. `1 + (a << 3) + ~a`
4. `a / 4`		d. `(a << 4) + (a << 2) + (a << 1)`
5. (a < 0) ? 1 : -1		e. `((a < 0) ? (a + 3) : a) >> 2`
		f. `a ^ (INT_MIN + INT_MAX)`
		g. `~((a | (~a + 1)) >> W) & 1`
		h. `~((a >> W) << 1)`
		i. `a >> 2`

vraag 2

a. Leg uit wat het verschil is tussen een Harvard en een Van Neuman machine.

b. Zowel de *Single* as *Multi Cycle* machine die jullie bij Digitale Methoden practica bestudeerd hebben hadden een *sign extender* component. Leg uit waarom deze component noodzakelijk is voor deze machine gebruikte getal representaties.

c. Op de computer systemen zijn gehele getallen gerepresenteerd in een tweetallig stelsel. Men kan getallen natuurlijk ook in drietallig stelsel representeren, bijvoorbeeld met de functie:

$$B2D_w(\vec{x}) = \sum_{i=0}^{w-1} x_i 3^i$$

Schrijf 24 en 31 in het bovenstaande (*unsigned*) drietallig stelsel op en tel de getallen in het drietallig stelsel bij elkaar op.

Student: **Collegekaartnummer:**

vraag 3

De volgende code wordt uitgevoerd op een machine waar het type `int` een 32-bit *two's component* representatie heeft. Variabelen van het type `float` en `double` worden op deze machine respectivelijk met het 32-bit en 64-bit *IEEE format* gerepresenteerd. U kunt de bijzonderheid van de IA32 architectuur (die intern een 80 bit representatie gebruikt) voor deze opgave even vergeten.

In de code worden eerst willekeurige *integer* waarden gegeneerd, die vervolgens naar het type `double` worden geconverteerd:

```
1      /* Create some random values */
2      int x = random();
3      int y = random();
4      int z = random();
5      /* convert to double */
6      double dx = (double )x;
7      double dy = (double )y;
8      double dz = (double )z;
```

Geef voor de volgende stukjes C-code aan of de test **in alle gevallen** waar is, of niet. Indien naar u mening **waar**, dient u dat te ondersteunen met een afleiding uit de onderliggende wiskundige eigenschappen. Indien naar u mening **niet** (in alle gevallen) **waar**, dient u dit aan te tonen door een voorbeeld te geven wanneer de test een 0 oplevert:

```
A      (double )(float )x == dx
B      dx + dy == (double )(x+y)
C      dx + dy + dz == dz + dy + dx
D      dx * dy * dz == dz * dy * dx
E      dx / dx == dy / dy
```

Student: **Collegekaartnummer:**

vraag 4

U heeft een re-engineering taak gekregen: u dient de C-code te reconstrueren die hoort bij de volgende definities, en de IA32 *assembly* code die ontstaan is bij het compileren van de oorspronkelijk C code.

```
struct s1 {
   char a[3];
   union u1 b;
   int c;
};
```

```
struct s2 {
   struct s1 *d;
   char e;
   int f[4];
   struct s2 *g;
};
```

```
union u1 {
   struct s1 *h;
   struct s2 *i;
   char j;
};
```

Zoals u weet is de grootte van een `char` 1 byte, en een `int` 4 bytes Het is misschien handig om configuratie van de drie data-structuren hieronder te tekenen:

Student: **Collegekaartnummer:**

Vul nu voor elk van de vier C-functies aan de rechterkant de missende code in, aan de hand van de *assembly-code* aan de linkerkant. De instructie `movsbl` kopieert een enkele byte naar een 32 bits adres, en vult de andere 3 bytes op met de *sign*-bit van de gekopieerde byte:

A.
```
proc1:
    pushl %ebp
    movl %esp,%ebp
    movl 8(%ebp),%eax
    movl 12(%eax),%eax
    movl %ebp,%esp
    popl %ebp
    ret
```
```
int proc1(struct s2 *x)
{
  return x->____________________ ;
}
```

B.
```
proc2:
    pushl %ebp
    movl %esp,%ebp
    movl 8(%ebp),%eax
    movl 4(%eax),%eax
    movl 20(%eax),%eax
    movl %ebp,%esp
    popl %ebp
    ret
```
```
int proc2(struct s1 *x)
{
  return x->____________________ ;
}
```

C.
```
proc3:
    pushl %ebp
    movl %esp,%ebp
    movl 8(%ebp),%eax
    movl (%eax),%eax
    movsbl 4(%eax),%eax
    movl %ebp,%esp
    popl %ebp
    ret
```
```
char proc3(union u1 *x)
{
  return x->____________________ ;
}
```

D.
```
proc4:
    pushl %ebp
    movl %esp,%ebp
    movl 8(%ebp),%eax
    movl (%eax),%eax
    movl 24(%eax),%eax
    movl (%eax),%eax
    movsbl 1(%eax),%eax
    movl %ebp,%esp
    popl %ebp
    ret
```
```
char proc4(union u1 *x)
{
  return x->____________________ ;
}
```

Succes!