# ICP Midterm Solution

1 a) Infinite loop, since the value of **`USHRT_MAX+1`** is defined to be 0.

b) The behavior of this loop is undefined, since the value of **`SHRT_MAX +1`** is undefined. Usually, it will result in an infinite loop, too.

2 a)
```
2
NT$23,456.00
13
```
b)
```
177776 fffffffe
```
c)
```
66 B
```

3 a) value: **`UINT_MAX`**; type: **`unsigned`**; # of type conversions: 1

It is equivalent to
```
1==1? (unsigned)-1: 1u
```

b) value: **`4.0f`**; type: **`float`**; # of type conversions: 5

It is equivalent to
```
(float)(2*(int)((int)'1'>(int)'0'))+2.0f*(float)true
```

4 a)
```
i<n&&a[i]!=k
```
// watch the order

b)
```
m<a[i]? a[i]: m
```
or
```
m<=a[i]? a[i]: m
```

5 a) 13579

b) 1253164297531

6 a) **`m*n<=UINT_MAX`** is always true.

b) The expression returned should be
```
m==0u||n==0u? 0u: m<=UINT_MAX/n? m*n: 0u
```
or
```
m==0u||n==0u||m>UINT_MAX/n? 0u: m*n
```
For the latter, watch the order of the operands of the logical operators.

7 a)
```
406ccccd
```
b)
```
union { float x; int y; } z={3.7f};
printf("%x\n",z.y);
```

8 a) `{ s+=i; ss+=s; }` or `ss+=s+=i;`

b) `ss+=i*(n-i+1)`

9 a) ① `r>=10` ② `r>0` ④ `n>0`

Only ③ `n>=10` can't be used as the terminiation condition, since it causes the function to respond incorrectly on some inputs, e.g. 300023

b)
```
int j=0;
while (i>j&&d[i]==d[j]) { i--; j++; }
return !(i>j);
```
or
```
int j=0;
while (i>j)
    if (d[i]==d[j]) { i--; j++; }
    else return false;
return true;
```

10 a) For $n = 0$, the loop incorrectly sets `d[0]` = 0.
The correct setting should be `d[0]` = 1.

b) The do…while loop takes two more divisions than the while loop.
More precisely, suppose that $n$ has $k$ digits. Then,

| | # of comparisons | # of increments | # of divisions |
|---|---|---|---|
| while loop | $k$ | $k$ | $2(k-1)$ |
| do while loop | $k$ | $k$ | $2k$ |

11 a)
```
int j;
for (j=i;j>=1;j--) {
    int k;
    for (k=j-1;k>=0;k--)
        if (a[j]==a[k]) break;
    if (k>=0) break;
}
return j==0;
```

b)
```
bool GOOD=true;
for (int j=i;j>=1&&GOOD;j--) {
    for (int k=j-1;k>=0&&GOOD;k--)
        if (a[j]==a[k]) GOOD=false;
}
return GOOD;
```

12
```
int f(int m,unsigned n)
{
    int z=m+(int)sqrt(n);
    return z*z+3*z+5;
}
```

Poor solution

```
int f(int m,unsigned n)
{
    int z=m+sqrt(n);       // need one more conversion
    return z*z+3*z+5;
}
```

Poor solution

```
int f(int m,unsigned n)
{
    return (m+(int)sqrt(n))*(m+(int)sqrt(n))
                                    +3*(m+(int)sqrt(n))+5;
}
```