Fall 2009

ICP Midterm

Problems 1~8: 4% each

1 In order to use each symbolic constant or function below, which header file must be included?

a) **INT_MAX** b) **abs** c) **sqrt** d) **scanf**

2 Given

```
char a[bits];
```

and consider the following three ways to define **bits**

① **#define bits 32** ② **int bits=32;** ③ **const int bits=32;**

a) Which is (are) legal in C89?

b) Which is (are) legal in C++?

3 For each expression below, determine if it evaluates to true iff the ***signed*** integer **n** is odd.

① **n%2==1** ② **(n&1)==1** ③ **n>>1<<1!=n** ④ **abs(n)%2==1**

4 What is the type of the value of each expression below?

a) **sizeof(int)**

b) **1==1?1:1u**

5 Several implicit conversions occur in the following C definition

```
short x=sqrt(x+2)+x;
```

Rewrite the definition by making the implicit conversions explicit, i.e. using the cast operator.

6 Let **n** be an unsigned integer, write an expression in C to that evaluates to true if **n** is a multiple of 2 or 3, but not a multiple of 6.

7 Explain why the output of the following code is indeterminate.

```
printf("%d\n",printf("Snoopy")+printf("Pluto"));
```

8 There is an error in the following program. Find and correct the error.

```
#include <stdio.h>
int main(void)
{
   for (int x=1;x<=33;++x,x++) printf("%d",db(x));;
   return 0;
}
int db(int x) { return x+x; }
```

Problem 9: 12%

9 Fill in the following blanks

a) // return true iff there is a zero in the array **a** of **n** elements

```
bool zero(int a[],int n)
{
   int i; for (i=0;______;i++); return i<n;
}
```

b) // implement Euclid's algorithm, i.e. gcd(a,b) = a, if b = 0; = gcd(b,a mod b), if b > 0

```
unsigned gcd(unsigned a,unsigned b)
{
   while (b>0) ______
   return a;
}
```

c) // compute **a*b** using bitwise and additive operations

```
unsigned mul(unsigned a,unsigned b)
{
   unsigned r=0;
   for (int i=0;i<8*sizeof(b);i++)
      if (______) r+=a<<i;
   return r;
}
```

Problems 10~16: 8% each

10 Show the output of each code segment below, assuming that sizeof(int)=4 and sizeof(short)=2

a)

```
signed short n=-1;
printf("%hu %x",n,n);
```

b)

```
unsigned x=5,y=6;
printf("%u %u %u %u",x&&y,x&y,x|y,x^y);
```

11 Show the output of each code segment below.

(Show your work for partial credits, in case your answers are incorrect.)

a)

```
signed short n=-22;
printf("%ho",n);
```

b)

```
union { float f; int n; } x={-0.2f};
printf("%x",x.n);
```

12 Show the output of each loop below.

a)
```
for (int i=1;i<=7;i+=2) {
    for (int j=i;j<=7;j+=3) printf("*");
    printf("\n");
}
```

b)
```
int n=1234567,s=0;
while (n>0) {
    s=s*100+n%100%10*10+n%100/10;
    n/=100;
    printf("%d\n",s);
}
```

13 Consider the following two loops:

Loop A
```
while (n>0)
    if (n%2==0) n/=2; else n--;
```

Loop B
```
while (n>0) {
    while (n%2==0) n/=2;
    n--;
}
```

a) For `n` = 2^k, $k \geq 0$, how many times is each expression below evaluated in Loop A? In B?
① `n>0` ② `n%2==0`

b) Modify Loop B to count the number of times each expression mentioned in a) is evaluated.

14 Consider the following function given in the lecture to generate 0!, 1!, 2!, … until overflow.

```
void factgen(void)
{
    for (unsigned k=0;;k++) {
        unsigned r=1;
        for (unsigned i=2;i<=k;i++)
            if (r<=UINT_MAX/i) r*=i; else return;  //*
        printf("%u!=%u\n",k,r);
    }
}
```

a) Suppose the test `r<=UINT_MAX/i` in the starred line is replaced by `r*i<=UINT_MAX`, what would be the result of executing the call `factgen()`? **Explain.**

b) Suppose the `return` statement in the starred line is replaced by a `break` statement, the function fails. What else must be changed to make it work?
(You needn't rewrite the entire code. Just write down the necessary changes.)

15 Consider the following function of HW#2

```
int b(int n)
{
    int k=1,a=fib(k);
    while (n>=a) { n-=a; k++; a=fib(k); }
    return k;
}
```

where the call `fib(k)` computes the `k`$^{\text{th}}$ Fibonacci number.

a) Explain why the code is inefficient.

b) Rewite the code to improve its efficiency.
(You needn't rewrite the entire code. Just write down the necessary changes.)

Note: Recall that the function `b` computes the `n`th item of the sequence

$$\underbrace{1}_{F_1\,1},\ \underbrace{2}_{F_2\,2},\underbrace{3,3,}_{F_3\,3s}\underbrace{4,4,4,}_{F_4\,4s}\underbrace{5,5,5,5,5,}_{F_5\,5s}\underbrace{6,6,6,6,6,6,6,6,}_{F_6\,6s}\dots$$

where $F_1, F_2, F_3, F_4, F_5, F_6, \dots$ is the sequence of Fibonacci numbers 1, 1, 2, 3, 5, 8, …

16 Write a function

`bool distinct(int a[],int n);`

that returns true iff the `n` characters of array `a` are all distinct.

For example, given

`int a[8]={1,3,5,7,2,4,6,8};` // all elements are distinct

`int b[6]={1,2,3,5,4,3};` // 3 appears twice

then

`distinct(a,8)` is true, but `distinct(b,6)` is false.

Requirement (3 points)

DO NOT use `break` or `return` to exit a ***loop***.