Ayman Hajja. Sat Jul 15 2017 11:37:07 GMT-0400 (EDT)

****

1) Convert the decimal number: 1 to a 5-bit unsigned binary
00001

****

2) Convert the decimal number: 5 to a 5-bit unsigned binary
00101

****

3) Convert the decimal number: 28 to a 5-bit unsigned binary
11100

****

4) Convert the decimal number: 29 to a 5-bit unsigned binary
11101

****

5) Convert the decimal number: 13 to a 5-bit unsigned binary
01101

****

6) Convert the decimal number: -11 to a 5-bit 1's complement binary (if value can't be represented, answer 'NA')
10100

****

7) Convert the decimal number: 11 to a 5-bit 1's complement binary (if value can't be represented, answer 'NA')
01011

****

8) Convert the decimal number: 17 to a 5-bit 1's complement binary (if value can't be represented, answer 'NA')
NA

****

9) Convert the decimal number: 18 to a 5-bit 1's complement binary (if value can't be represented, answer 'NA')
NA

****

10) Convert the decimal number: -8 to a 5-bit 1's complement binary (if value can't be represented, answer 'NA')
10111

****

11) Convert the decimal number: -6 to a 5-bit 2's complement binary (if value can't be represented, answer 'NA')
11010

****

12) Convert the decimal number: 1 to a 5-bit 2's complement binary (if value can't be represented, answer 'NA')
00001

****

13) Convert the decimal number: -13 to a 5-bit 2's complement binary (if value can't be represented, answer 'NA')
10011

****

14) Convert the decimal number: 0 to a 5-bit 2's complement binary (if value can't be represented, answer 'NA')
00000

****

15) Convert the decimal number: 11 to a 5-bit 2's complement binary (if value can't be represented, answer 'NA')
01011

****

16) What will be the output of the following C program ?

```
#include <stdio.h>
int main()
{
int x = 17;
int *y;
int *z;
y = &x; /* Assume address of x is 500 (decimal) and size of integer is 4 byte long */
z = y;
*y = *z + 1;
x = x + 1;
printf("x = %d, y = %p, z = %p\n", x, y, z); // Use decimal when printing pointers
return 0;
}
```

x = 19, y = 500, z = 500

****

17) For a system of n-digit unsigned base 4 numbers (n > 1), how many numbers (unique combinations) can be represented?
4^n

****

18) For an n-digit 2's complement binary number (n > 1), what is the number of negative integers (as a function of n)?
2^(n-1) or (2^n)/2

****

19) For an n-digit 2's complement number (n > 1), how many zeros are there?
1

****

20) Write a 'swap' function with the following function header:

```
void swap(int *p1, int *p2);
```

The 'swap' function should swap the values of two integers.

```
int main()
{
  int x = 10;
  int y = 20;
  // You must figure out how to call the function correctly (include this in your answer)
  // Next line should print out x: 20, y: 10
  //
  printf("x: %d, y: %d\n", x, y);
}
void swap(int *p1, int *p2)
{
int temp = *p1;
*p1 = *p2;
*p2 = temp;
}
```

To call the function from main():
```
swap(&x, &y);
```

****

21) According to the C standard, arr[0] is actually syntactic shorthand for *(arr+0). Write a C program that loops twice, the first loop is to initialize the elements of some integer array (say size 20), and a second loop to print all the elements of the array (next to their addresses). In both loops, use the alternative notation (* notation).

The output of your code should look like the following:

```
<address of 1st element>, <value of 1st element> (e.g. 0x7fff5fbff63c, 50)
<address of 2nd element>, <value of 2nd element> (e.g. 0x7fff5fbff640, 50)
```

```
int main()
{
int my_array[20];
int i;

for (i = 0; i < 20; i ++)
*(my_array + i) = 50;

for (i = 0; i < 20; i++)
printf("%p, %d\n", my_array + i, *(my_array + i));

return 0;
}
```

****