COP 3502 Study Group Sheet: Base Conversion, Bitwise Operators Solutions

1) Convert 5346_7 to base 10.

Solution

$5346_7 = 5 \times 7^3 + 3 \times 7^2 + 4 \times 7^1 + 6 \times 7^0 = 1715 + 147 + 28 + 6 =$ **1896**

2) Convert 1926_{10} to base 12.

Solution

12 | 1926
12 | 160 R 6
12 | 13 R 4
12 | 1 R 1

$1926_{12} =$ **1146_{12}**

3) There are a total of 25 cards, numbered 0 through 24. We can represent a set of cards with a single integer by setting the i^{th} bit to 1 if the set contains card i, and setting the bit to 0 otherwise. For example, the set of cards {2, 6, 7} would be stored as the integer 196, since $196 = 2^7 + 2^6 + 2^2$. Two sets of cards are disjoint, if and only if no card appears in both sets. Complete the function below so that it returns 1 if the sets of cards represented by the integers set1 and set2 are disjoint, and returns 0 if they are not disjoint. (For example, disjoint(196, 49) should return 1 because $49 = 2^5 + 2^4 + 2^0$, and there is no common value in the two sets {2, 6, 7} and {0, 4, 5}. On the other hand, disjoint(196, 30) should return 0 because $30 = 2^4 + 2^3 + 2^2 + 2^1$, so that card number 2 is included in both sets 196 and set 30.)

```
// Pre-condition: set1 and set2 are bitmasks in between 0 and (1<<25)-1.
// Post-condition: Returns 1 if the two bitmasks are disjoint, meaning that the sets they represent
//       don't have any items in common, and returns 0 otherwise, if the two represented sets do
//       have common items.
int disjoint(int set1, int set2) {

    for (int i=0; i<25; i++) {
        if ( (set1&1) && (set2&1) ) return 0;
        set1 >>= 1;
        set2 >>= 1;
    }

    return 1;
}

int disjoint(int set1, int set2) {
    if ( (set1&set2) == 0) return 1;
    return 0;
}
```

4) In the game of NIM, there are several piles with stones and two players alternate taking 1 or more stones from a single pile, until there are no more stones left. The person who takes the last stone wins. It turns out that if it's someone's turn, if they play optimally, they can win as long as the bitwise XOR of all of the number of stones in each pile is not equal to 0. Write a function that takes in an array of values representing the number of stones in the piles of NIM and the length of that array, and returns 1, if the current player can win, and 0 otherwise, assuming both players play optimally.

```
int canWinNIM(int piles[], int numPiles) {

    int res = 0;
    for (int i=0; i<numPiles; i++)
        res ^= piles[i];

    return res != 0;
}
```

5) A company is looking to hire an employee based on answers to a questionnaire. The questionnaire has 20 yes/no questions (labeled from question 0 to question 19) and an applicant's set of responses is stored as a single integer such that the ith bit is set to 1 if the response to question i is yes, and it's set to 0 if the response to question i is no. For example, if an applicant answered yes to questions 0, 2, 3, 5, 8, and 10 and no to the other 14 questions, her responses would be stored as the single integer 1325, since 00000000101001011012 = 1325. For all questions, the company prefers yes answers to no answers. However, since it's unlikely that a candidate will answer yes to all of the questions, they have developed a modified scoring system for each candidate. The company has ranked each of the 20 questions in order of preference from most important to least important. This ranking is an array, ***preferences***, that stores a permutation of the integers from 0 to 19, inclusive. For example, if preferences[0] = 8, preferences[1] =10 and preferences[2] = 1, then the company cares most about the answer to question 8, second most about the answer to question 10 and third most about the answer to question 1. A candidate's score is simply the maximum number of the most important questions they answered yes without a single no. For example, the sample candidate described above would have a score of 2, because she said yes to questions 8 and 10, but no to question 1, which was third on the preference list. Any candidate who said no to question 8 would be assigned a score of 0. Complete the function below so it returns the score of the applicant whose answers are stored in the formal parameter ***applicant***. The formal parameter ***preferences*** is an array of size 20 which stores the order of importance of the questions as previously described.

```
#define SIZE 20
int score(int preferences[], int applicant) {

    int res = 0;
    for (res=0; res<SIZE; res++)
        if ( (applicant & (1<<preferences[res])) == 0)
            break;

    return res;
}
```