Name: _____ Tutorial Section: 02 03 04 05 06 09

Important Note: Please write your student number on the back of this page.

APS 105 QUIZ # 3 Computer Fundamentals

Fall 2010 SECTION: 2-3PM Tuesday, November 9

All quizzes, tests, and exams in this course are **closed book** with **no calculators**.

[10 Marks] Write a function named `diag` having prototype `int **diag (int n, int k)` that returns a dynamically allocated 2-dimensional array that can hold a `n`×`n` matrix of `int`. The dynamically allocated 2-dimensional array should be created using call(s) to the `malloc` function. Before returning the array, the `diag` function should zero all of the elements and then place the value 1 on the `k`-th diagonal of the array. The value `k = 0` is used to represent the main diagonal of the array. If `k > 0`, the diagonal that is to contain the 1's is `k` diagonals **above** the main diagonal. If `k < 0`, then the diagonal that is to contain the 1's is `-k` diagonals **below** the main diagonal.

For example, the function calls `diag(4,0)`, `diag(4,1)` and `diag(4,-2)` should return arrays holding the matrices

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

respectively.

Write your `diag` function on the back of this page.

Student Number: _____

Sample solution:

```
int **diag (int n, int k)
{
    int r,c;
    int **A = (int **)malloc(n*sizeof(int *));
    for (r=0; r<n; r++)
    {
        A[r] = (int *)malloc(n*sizeof(int));
    }

    for (r=0; r<n; r++)
    {
        for (c=0; c<n; c++)
        {
            A[r][c] = 0;
        }
    }

    if (k >= 0)
    {
        for (r=0; r<n-k; r++)
        {
            A[r][r+k] = 1;
        }
    }
    else
    {
        k = -k;
        for (r=k; r<n; r++)
        {
            A[r][r-k] = 1;
        }
    }

    return A;
}
```