

NAME \_\_\_\_\_ SS# \_\_\_\_\_

**To receive maximum partial credit, clearly show all of your work.**

1. **(Functions)** Write a function to perform division. The function is to accept 3 doubles, the numerator and the denominator and a pointer to the remainder. The function is to divide the 2 doubles and produce a quotient and a remainder. The quotient is to be returned through the function name and the remainder through the pointer in the third parameter. Assume you have the functions `floor ( )` which returns the integer part of a double and the function `frac( )` which returns the fractional part of a double.

Algorithm:

$$\text{Result} = \text{numerator} / \text{denominator};$$
$$\text{Quotient} = \text{floor}(\text{result})$$
$$\text{Remainder} = \text{frac}(\text{result}) * \text{denominator}$$

2. **(File)** Write a program to open two files called “input1.dat” and “input2.dat” both for input. Read the first 5 integers from each file and compute the sum of the 10 integers. Print the sum then close the files. Print an error message if the file cannot be opened.

3. **(Classes)** Create a class called Point that has as private elements the x and y positions on the Cartesian graph. The class is to have as public the color code of the point represented by an integer, a function that returns the distance the point is from the origin, use the formula  $dist = \sqrt{x^2 + y^2}$ , and a constructor that initializes the point's x and y position and sets the color to 0.

4. (**Scope**). What is the output if you run the program 3 times and each time you enter 1,2 and 3 for k.

```
int z;
void main()
{
    int    *p,*q,x,y,k,      (k == 1)      _____
           w[3] = {10,20,30};
    char   c;

    p = &x;
    x = 25;                    (k == 2)      _____
    y = 36;
    z = 47;
    q = &(w[1]);

    scanf("%d", &k);          (k == 3)      _____

    if (k == 1)
        {
            int x;

            x = 60;
            y = 75;
            z = 82;
        }
    else if (k == 2)
        {
            int x;

            *p = 3;
            x = 6;
            y = *p + *p;
            z = x + y;
        }
    else
        {
            x = 71;
            y = 28;
            z = 39;
            *(q + 1) = 90;
        }
    printf ("x=%d, y= %d z= %d", x, y, z);
    printf (" w[0] = %d w[1] = %d w[2]=%d \n", w[1], w[2], w[3]);
}
```

5. (pointers and arrays) On the right side show what is in the array.

```
main()
{
  int    a[2][3],b[5],*p;

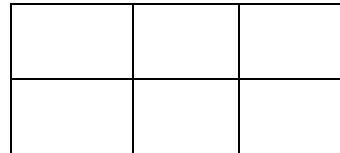
  p = b;
  *p = 4;
  p = p + 3;
  p[1] = 1;
  b[3] = 2;
  *(p - 2) = 3;

  a[1][0] = 1
  *(a + 4) = 2;
  p = a + 3;
  *(p + 2) = 3;
  p = &(a[0][2]);
  *(p - 1) = 4;
}
```

b



a



6. (Short answers). Answer each of the following.

Assume  $a = 3$ ,  $b = 7$ ,  $c = 17$ .

```
for (x = -1; x < 2; x++)
    for (y = 0; y < 2; y++)
        printf("%d %d\n",x,y); _____
```

```
for (x = 0; x < 8; x = x + 3)
{
    y = 1;
    while ( y < x )
    {
        printf(" %d %d \n ", x, y);
        y = y * 2;
    }
} _____
```

```
for (x = 0; x < 6; x++)
    for (y = 0; y < 10; y++)
        if (x + y == 10)
            printf("%d %d\n",x,y); _____
```

```
if ( ( a == 3 && b < 10 ) || ( a == 4 && b > 10 ) )
    printf("Red \n");
else
    printf("Blue \n"); _____
```

```
if ( a == 3 || b < 4)
    if (b == 5 && c < 10)
        printf("Yellow \n");
    else
        printf("Green \n");
else
    if (b == 15 && c > 10)
        printf ("Orange \n");
    else
        printf("Black \n"); _____
```

