COSC 120 **Exam 2 Key** Spring 2012

Part 1: Short Answer (5 Points Each)

1. Write a declaration that creates an array of integers `arr` and loads it with the values 3, 4, 2, 7, 6, 19, 23, and 21. This is to be done in one line all in the declaration.
```
int arr[] = {3, 4, 2, 7, 6, 19, 23, 21};
```

2. Give a brief description of what *passed by pointer* means. --- Arrays are passed by pointer, which means that the memory address of the beginning position of the array is passed to a function. It is similar to being passed by reference in that changes to the array are visible in the calling function. One can lock the array from being changed by using the const keyword in the parameter list.

3. What unexpected side effect might happen in the following block of code and why would it happen?

```
int A[10];

for (int j = 0; j < 15; j++)
    A[j] = j;

for (int j = 0; j < 15; j++)
    cout << A[j] << "  ";
```

   Since the array is declared as size 10, only 10 locations are "reserved". Since the for loops store and retrieve 15 integers it is possible that the last 5 are corrupted.

4. Write a typedef statement that will allow the user to use the data type `Jack` in place of a double array with 100 elements in it.
```
typedef double Jack[100];
```

5. Give a short description of what the keyword `static` does. --- When the keyword static is used on a variable in a function the variable is created and initialized and used as any other variable in the function but it is not destroyed when the function ends. So if the function is called again the value of the variable is retained and can be used without reinitializing it.

6. Write the declaration of a two-dimensional array of integers where the array would have 100 rows and 200 columns.
```
int A[100][200];
```

7. Write a function header for a value-returning function, which returns a boolean, called `DoThis` that takes in a one-dimensional array of long integers as a parameter and allows the array to be changed.
```
bool DoThis(long A[])
```

Part 2: Program Traces (15 points Each)

1. For each of the inputs in the box on the right give the output of the following program.

```
#include <iostream>

using namespace std;

int main()
{
    int A[5];
    int m;

    cin >> m;

    for (int i = 0; i < 5; i++)
        A[i] = i;

    for (int i = 0; i < 5; i++)
    {
        int index = ((5*i+A[i]) % m) % 5;
        int temp = A[index];
        A[index] = A[i];
        A[i] = temp;
    }

    for (int i = 0; i < 5; i++)
        cout << A[i] << "  ";

    cout << endl;

    return 0;
}
```

```
3
4   0   1   2   3


5
0   1   2   3   4


7
2   1   3   4   0
```

2. For each of the inputs in the box on the right give the output of the following program.

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
    int A[5];
    int B[5];
    int C[10];

    for (int i = 0; i < 5; i++)
        cin >> A[i];

    for (int i = 0; i < 5; i++)
        cin >> B[i];

    for (int k = 0; k < 5; k++)
    {
        C[2*k] = A[k];
        C[2*k+1] = B[k];
    }

    for (int i = 0; i < 9; i++)
    if (C[i] > 5)
    {
        int temp = C[i];
        C[i] = C[i+1];
        C[i+1] = temp;
        i++;
    }

    for (int i = 0; i < 10; i++)
        cout << C[i] << "  ";

    cout << endl;

    return 0;
}
```

```
1  2  3  4  5  6  7  8  9  0

1  2  6  3  7  4  8  5  9  0



7  2  9  5  6  3  1  0  8  4

3  7  2  1  0  9  5  6  8  4
```

Part 3: Coding (15 points Each)

1. Write a program that will read in integers from a file into an array. The number of integers is unknown but we know that there are less then 2000 of them. Write a value-returning function that will take the array as a parameter and output the number of integers in the array that are larger then 100. Have the main print this number out.

```
#include <iostream>
#include <fstream>

using namespace std;

int countBig(int A[], int size)
{
    int cnt = 0;
    for (int i = 0; i < size; i++)
        if (A[i] > 100)
            cnt++;

    return cnt;
}

int main()
{
    int numbers[2000];
    int arraySize = 0;
    int num = 0;
    ifstream dataFile;
    char ch;

    dataFile.open("numbers001.txt");

    if(!dataFile)
    {
        cout << "Error opening file.\n";
        cin.ignore();
        return 1;
    }

    while ((ch = dataFile.peek()) != EOF)
    {
        dataFile >> num;
        numbers[arraySize] = num;
        arraySize++;
    }
    dataFile.close();

    cout << countBig(numbers, arraySize) << endl;

    return 0;
}
```

2. Below we have the function header for a Linear Search function. Complete the code so that the function will do a linear search on the array of size, size and determine if the target is in the array. If the target is in the array return the index of the target and if the target is not in the array have the function return –1.

```
int LinearSearch(int arr[], int size, int target)
{
    for (int i = 0; i < size; i++)
        if (arr[i] == target)
            return i;
    return -1;
}
```

3. Below we have the function header for a Selection Sort function. Complete the code so that the function will do a selection sort of the array. The size of the array is size.

```
void SelectionSortArray(int list[], int size)
{
    int pos;
    int minIndex;
    int minValue;
    for (pos = 0; pos < (size-1); pos++)
    {
        minIndex = pos;
        minValue = list[pos];
        for(int index = pos + 1; index < size; index++)
        {
            if(list[index] < minValue)
            {
                minValue = list[index];
                minIndex = index;
            }
        }
        list[minIndex] = list[pos];
        list[pos] = minValue;
    }
}
```