FINAL CS111 Fall 2019 TEST B

Instructions:

1) Make sure your name and CUNY ID are filled in.

2) For programming questions, in most cases, I have written the signature of a function for you, and you just have to fill in the rest of the function.

3) You can only use functions: a)that are used in this test (as questions) b) ones that you write yourself or c) functions that are defined below.

The only exception is that you can use the cout, cin, rand, and the time functions.

```
void Swap(char &c1, char &c2) { char temp = c1; c1 = c2; c2 = temp; }
bool IsLower(char c) { return(c >= 'a'&&c <= 'z'); }
bool IsUpper(char c) { return(c >= 'A'&&c <= 'Z'); }
char ToLower(const char c) { return(IsUpper(c) ? 'a' + c - 'A' : c); }
char ToUpper(const char c) { return(IsLower(c) ? 'A' + c - 'a' : c); }
bool IsAlpha(char c) { return(IsLower(c) || IsUpper(c)); }
bool IsNumeric(char c) { return(c >= '0'&&c <= '9'); }
bool IsAlphaNumeric(char c) { return(IsAlpha(c) || IsNumeric(c)); }
bool IsSpace(char c) { return(c == ' ' || c == '\t' || c == '\r' || c == '\n'); }
void MakeUpper(char &c) { c = ToUpper(c); }
void MakeLower(char &c) { c = ToLower(c); }
bool IsSame(const char c1, const char c2) { return(ToUpper(c1) == ToUpper(c2)); }
int StringLength(const char *str) { int i = 0; while (str[i])i++; return(i); }
int Count(const char *str, const char c) { int i = 0; int count = 0; while (str[i]) { if (str[i] ==
c)count++; i++; }return(count); }
bool Contains(const char *str, const char c) { return(Count(str, c) > 0); }
char * StringCopy(char *dest, const char *src, const int max = 1000000) { int i = 0; while (src[i] && i <
(max - 1)) { dest[i] = src[i]; i++; }dest[i] = 0; return(dest); }
int IndexInArray(const int *nums, const int size, const int num) { for (int i = 0; i < size; i++)if
(nums[i] == num)return(i); return(-1); }
int Absolute(const int num) { return(num < 0 ? -num : num); }
void Swap(int &i1, int &i2) { int temp = i1; i1 = i2; i2 = temp; }
bool HasDecimal(double d) { int i = d; return(i != d); }
int Max(const int *nums, const int size) { int max = nums[0]; for (int i = 1; i < size; i++)if (max <
nums[i])max = nums[i]; return(max); }
int Min(const int *nums, const int size) { int min = nums[0]; for (int i = 1; i < size; i++)if (min >
nums[i])min = nums[i]; return(min); }
int Sum(const int *nums, const int size) { int sum = nums[0]; for (int i = 1; i < size; i++)sum +=
nums[i]; return(sum); }
double Sum(const double *nums, const int size) { double sum = nums[0]; for (int i = 1; i < size; i++)sum
+= nums[i]; return(sum); }
int Mode(const int *nums, const int size) { int count[101]; for (int i = 1; i < 101; i++)count[i] = 0; for
(int i = 0; i < size; i++)count[nums[i]] ++; int most = -1; for (int i = 1; i < 101; i++) { if (count[i] >
most)most = i; }return(most); }
bool Contains(const int *answer, const int size, const int searchFor) { for (int i = 0; i < size; i++)if
(answer[i] == searchFor)return(true); return(false); }
bool IsSpecialCharacter(char c) { return(c == '!' || c == '@' || c == '#' || c == '~' || c == '%'); }
int * BubbleSort(int *nums, const int size) { bool bSwapped = true; while (bSwapped) { bSwapped = false;
for (int i = 0; i < size - 1; i++)if (nums[i] > nums[i + 1]) { Swap(nums[i], nums[i + 1]); bSwapped =
true; } }return(nums); }
int NumberOfDigits(const int i) { int num = i; int numOfDigits = 0; do { num /= 10; numOfDigits++; } while
(num); return(numOfDigits); }
void PrintArray(const int *arr, const int size) { int bufSize = NumberOfDigits(Max(arr, size)) + 1; for
(int i = 0; i < size; i++) { if (i) { for (int j = 0; j < bufSize - NumberOfDigits(arr[i - 1]); j++)cout
<< " "; }cout << arr[i]; }cout << endl; }
bool StringCompare(const char *str1, const char *str2){int i = 0;while (str1[i] && str1[i] ==
str2[i])i++;return(str1[i] == str2[i]);}
```

SECTION 1 – Miscellaneous Questions 5 Points Each

Q1 – What is the output of f1()?

Q1) _ A B _____

```
void f1()
{
	char str[10] = "ABCDEFGH";
	str[5 / 2] = 0;
	int i = 0;
	while (str[i])
	{
		cout << str[i] << " ";
		i++;
	}
	cout << endl;
}
```

Q2 – What is the output of f2()?

Q2) _ d g j m _____

```
void f2()
{
	for (int i = 0; i < 10; i += 3)
	{
		char c = 'd' + i;
		cout << c << " ";
	}
	cout << endl;
}
```

Q3 – What is the output of f3()?

Q3) _1000_____

```
void f3()
{
	int num = 1.2345;
	for (int i = 0; i < 3; i++)
	{
		num *= 10;
	}
	cout << num << endl;
}
```

Q4 – What is the output of f4()?

Q4) _ A B C_____

```
void f4()
{
	char arr[3][3] = {
		{'A','B','C'},
		{'D','E','F'},
		{'G','H','I'} };
	for (int i = 0; i < 3; i++)
		for (int j = 0; j <1; j++)
			cout << arr[j][i] << " ";
	cout << endl;
}
```

SECTION 2 – Basic Programming – Program the three function below. 10 Points Each

Q1 – RandomDigit – Returns a random digit, e.g. a number from 100 through 200.
Sample Output:
RandomDigit() ➔ 100
RandomDigit() ➔ 108

```
// whole number between 100 and 200 (inclusive)
int RandomDigit()
{
        return(100 + rand() % 101);
}
```

Q2 - Min - Returns the smallest number in an array. We look in the array at numbers that are after the start from startFrom position.
Sample Output:
int nums[10] = { 1,2,3,4,5,6,7,8,9,10 };
Min(nums,10,2)➔3
Min(nums, 10) ➔ 1

```
int Min(int *nums, const int size, const int startFrom=0)
{
        int min = nums[startFrom];

        for (int i = startFrom + 1; i < size; i++)
                if (min > nums[i])
                        min = nums[i];
        return(min);
}
```

Q3 - Average – Returns average of numbers in array. Numbers with a 0 value are ignored.
Sample Output:
int nums[4] = { 1,2,3,4};
Average(nums, 4) << endl;➔ 2.5
int nums2[3] = { 0,0,0 };
Average(nums2, 3) << endl;➔ 0
int nums3[3] = { 2,0,4 };
Average(nums3, 3) << endl;➔ 3

double Average(int *nums, const int size)

```
{
        int count = 0;
        double sum = 0.0;

        for (int i = 0; i < size; i++)
        {
                if (nums[i])
                {
                        count++;
                        sum += nums[i];
                }
        }
        return(count > 0 ? sum / count : 0);
}
```

SECTION 3 –Programming – Program the following functions 10 Points Each Choose 4 from the six below.

Q1) **Subset** - Returns true if all members of subset are also members of set
Suggestion - Use the Contains function
Sample Output:
int nums[10] = { 1,2,3,4,5,6,7,8,9,10 };
int nums2[5] = { 3,4,5,11,12};
int nums3[3] = { 3,4,5};
bool Subset(nums, 10,nums2,5) ➔ false
bool Subset(nums, 10,nums3,3) ➔ true

```
bool Subset(const int *set, const int setSize,
        const int *subset, const int subsetSize)
{
        for (int i = 0; i < subsetSize; i++)
                if(!Contains(set,setSize,subset[i]))
                        return(false);
        return(true);
}
```

Q2 – Intersect -This function outputs the characters that two arrays have in common.
Sample Output:
int nums[10] = { 1,2,3,4,5,6,7,8,9,10 };
int nums2[5] = { 3,4,5,11,12 };
int nums3[3] = { 10,11,12 };
Intersect(nums,10,nums2,5)➔ 3 4 5
Intersect(nums,10,nums3,3)➔ 10

```
void Intersect(const int *i1, const int iSize1,
        const int *i2, const int iSize2)
{
        for (int i = 0; i < iSize1; i++)
                if (Contains(i2, iSize2, i1[i]))
                        cout << i1[i] << " ";
        cout << endl;
}
```

Q3) MathArray

1) Fills the first two rows of the numbers array with random numbers.
2) Row 3 – 5 are the addition of the two rows above them.
3) Print out the array.
Steps 1) and 3) have already been done. You just have to complete step 2.
Sample Output:

Row 1 ==> 3 3 2 2 3
Row 2 ==> 2 1 1 2 3
Row 3 ==> 5 4 3 4 6
Row 4 ==> 7 5 4 6 9
Row 5 ==> 12 9 7 10 15

```
void MathArray()
{
	int numbers[5][5];

// STEP 1
	for (int i = 0; i < 2; i++)
		for (int j = 0; j < 5; j++)
			numbers[i][j] = (rand() % 3)+1;

// STEP 2
	for (int i = 2; i < 5; i++)
		for (int j = 0; j < 5; j++)
			numbers[i][j] = numbers[i - 1][j] + numbers[i - 2][j];



// STEP 3
	for (int i = 0; i < 5; i++)
	{
		for (int j = 0; j < 5; j++)
			cout << numbers[i][j] << " ";
		cout << endl;
	}
}
```

Q4 – Duplicates – This function writes all duplicates in the same array
Sample Output:
int nums[10] = { 4,2,3,4,5,6,6,8,9,10 };
int nums2[5] = { 3,4,5,11,12 };
int nums3[3] = { 1,1,1 };
Duplicates(nums,10➔4 6
Duplicates(nums2,5)➔
Duplicates(nums3,3) ➔ 1 1 1

```
void Duplicates(const int *nums, const int size)
{
	for (int i = 0; i < size; i++)
	{
		for (int j = i + 1; j < size; j++)
		{
			if (nums[i] == nums[j])
				cout << nums[j] << " ";
		}
	}
	cout << endl;
}
```

Q4 -ReverseOfEachOther – Compares two strings to see if they are the reverse of each other.
Sample Output:
ReverseOfEachOther("","") ➔ true
ReverseOfEachOther("ABC","CBA") ➔ true
ReverseOfEachOther("ABCDE","FECBA") ➔ false
ReverseOfEachOther("ABCDE","DCBA") ==> false

```
bool ReverseOfEachOther(const char *str1, const char *str2)
{
	if (StringLength(str1) != StringLength(str2))
		return(false);

	for (int i = 0,j= StringLength(str2)-1; i < StringLength(str1); j--,i++)
	{
		if (str1[i] != str2[j])
			return(false);
	}
	return(true);
}
```

Q5 - RandomString – Returns a random string of characters 0..1 a..z and A..Z using **size**
Sample Output:
RandomString(str, 5) ➔"s2aIO"
RandomString(str, 1) ➔"h"
RandomString(str, 10) ➔"oLe9Z3QXoS"

```
char *RandomString(char *str,int size)
{
	srand(time(0));

	for (int i = 0; i < size; i++)
	{
		int choice = rand() % 3;
		str[i] = choice == 0 ? '0' + rand() % 10 :
			(choice == 1 ? 'A' + rand() % 26 :
				'a' + rand() % 26);
	}
	str[size] = 0;
	return(str);
}
```

Q6 - WhenIsThanksgiving – In the USA, Thanksgiving is always celebrated on the fourth Thursday of November. This function initializes two arrays, 1) the days of November 2) The day of the week these days occur on. Using these arrays, calculate the day when thanksgiving occurs and using cout, output the date for thanksgiving.
Sample Output:
WhenIsThankgiving2019()➔11/28/2019

```
enum DayOfWeek
{
sun,mon,tue,wed,thu,fri,sat
};

void WhenIsThankgiving2019()
{
	DayOfWeek e = fri;
	DayOfWeek dayOfWeek[30];
	int i = 0; while (i < 30)
	{
		dayOfWeek[i] = e;
		i++;
		e = (DayOfWeek)(e + 1);
		if (e > sat)
			e = sun;
	}
	const char novDays[30][11] =
	{ "11/01/2019","11/02/2019","11/03/2019","11/04/2019","11/05/2019","11/06/2019",
	"11/07/2019","11/08/2019","11/09/2019","11/10/2019","11/11/2019","11/12/2019",
	"11/13/2019","11/14/2019","11/15/2019","11/16/2019","11/17/2019","11/18/2019",
	"11/19/2019","11/20/2019","11/21/2019","11/22/2019","11/23/2019","11/24/2019",
	"11/25/2019","11/26/2019","11/27/2019","11/28/2019","11/29/2019","11/30/2019" };

	int thursdayCount = 0;
	for (int i = 0; i < 30; i++)
	{
		if (dayOfWeek[i] == thu)
			thursdayCount++;
		if (thursdayCount == 4)
		{
			cout << novDays[i] << endl;
			break;
		}
	}
}
```