THE UNIVERSITY OF THE WEST INDIES
ST. AUGUSTINE

EXAMINATIONS OF DECEMBER 2016

Code and Name of Course: COMP1603 — Computer Programming III Paper:

Date and Time: Wednesday 21st December 2016 9 am Duration: 2 Hours

INSTRUCTIONS TO CANDIDATES: This paper has 5 pages and 3 questions

Answer all questions

1. (a) Consider the program shown below. Give the output of the program.

```
#include <iostream>
using namespace std;
int main() {
    void test (int *ptr, int n);
    int num[5];
    for (int j = 1; j < 5; j++)
       num[j] = 10 * j;
    test(num, 5);
    for (int j = 0; j < 5; j++)
      cout << "num[" << j <<"] is " << num[j] << endl;
    system ("pause");

    return 0;
}


void test(int *ptr, int max) {
   for (int j = 0; j < max; j++) {
      (*ptr) += 1;
       ptr++;
   }
 }
```

[5]

[Question 1 continues on the following page]

(b) (i) Write code to output the integers from 200 to 400 (in steps of 10) i.e. 200, 210, 220,….400 to a binary file "data.dat". [3]

(ii) Write code to read the values from "data.dat" above and place them in a stack, **s**. Next, pop the stack values one at a time and print only the values that are multiples of 20. You may assume the existence of the usual stack functions. Some prototypes are listed below.

```
Stack * initStack();
bool isEmpty (Stack * s);
bool isFull ( Stack * s );
int peek (Stack * s);
void push (Stack * s, int n);
int pop (Stack * s);
```
[7]

Total Marks for Question 1 is 15

2. (a) What output is produced by the call fun(18, 5) of the following recursive function? Show your working. [6]

```
void fun(int m, int n){
 if (n <= 0)
      cout << endl;
 else {
      fun(m - 2, n - 2);
      cout << m << " ";
      fun(m + 2, n - 2);

 }
}
```

(b) Write a **recursive** function to accept a positive integer **n** where n > 0 and print the digits of **n** in reverse order. For example, given 5678, the function prints 8765. [4]

[Question continues on the next page]

(c) Assume that **top** points to a linked list of integers that has already been created. The declarations for the nodes of the linked list follow:

```
struct Node {
   int data;
   Node * next;
};
```

Write a **recursive** function to find the sum of the data values in the linked list. The function prototype is

```
int recSum (Node *top);
```
[5]

Total Marks for Question 2 is 15

3. This question is based on linked lists of integers. The declarations for the nodes of the linked list follow:

```
struct Node {
   int data;
   Node * next;
};
```

(a) Write a function `mergeSorted` which accepts two pointers to linked lists sorted in ascending order. The function merges the two linked lists to form a new linked list also sorted in ascending order. The prototype for the function is

```
Node *mergeSorted (Node *list1, Node *list2)
```

Note that `list1` and/or `list2` may be empty initially.
Assume that there are no duplicates in `list1` and `list2`. [10]

(b) Write an efficient function, `containsSorted` that accepts a pointer to the top of a linked list (sorted in ascending order) and an integer `key`. The function returns `true` if key is found in the list and `false` otherwise. The function prototype is

```
bool containsSorted (Node *top, int key)
```
[6]

[Question continues on the next page]

(c) Write a function `removeDuplicates`, which takes a linked list sorted in **ascending order** as input and deletes any duplicate nodes from the list. Return the new list after the duplicates have been removed.

The function prototype is

```
Node *removeDuplicates(Node *top)
```

[14]

Total Marks for Question 3 is 30

End of Question Paper (Total Marks 60)