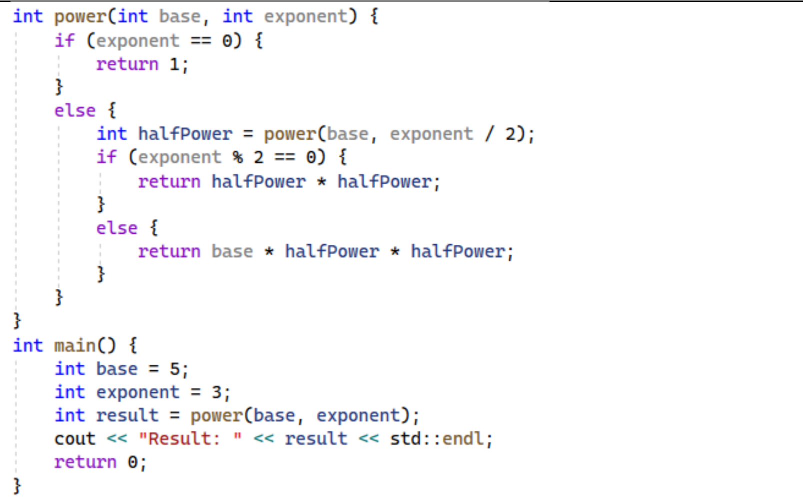
Data Structure Quiz 5

Total Marks 18

Name: ________________________________ Roll No. ______________________________

Question1: write output of the following codes. (8 marks)

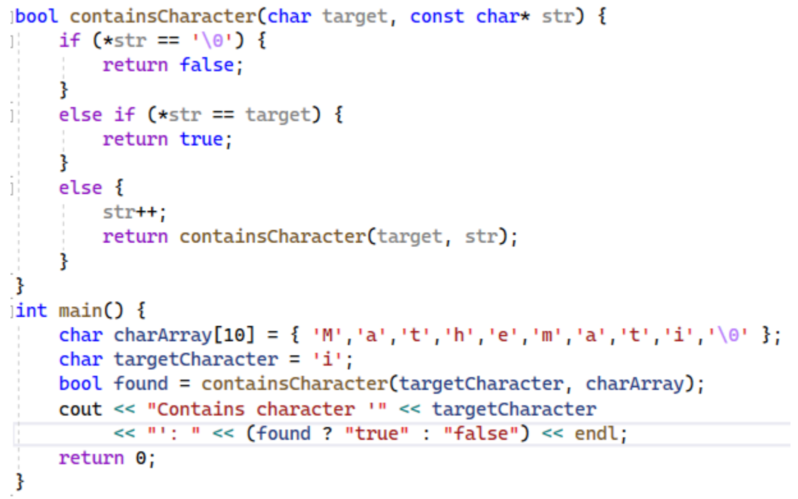
1)



```
int power(int base, int exponent) {
    if (exponent == 0) {
        return 1;
    }
    else {
        int halfPower = power(base, exponent / 2);
        if (exponent % 2 == 0) {
            return halfPower * halfPower;
        }
        else {
            return base * halfPower * halfPower;
        }
    }
}
int main() {
    int base = 5;
    int exponent = 3;
    int result = power(base, exponent);
    cout << "Result: " << result << std::endl;
    return 0;
}
```

Output:
125

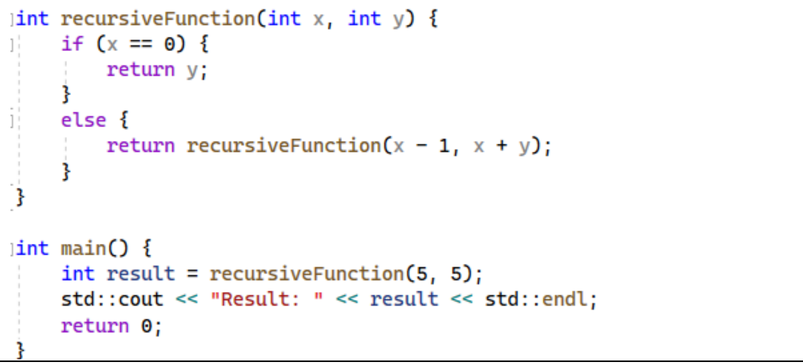
2)



```
bool containsCharacter(char target, const char* str) {
    if (*str == '\0') {
        return false;
    }
    else if (*str == target) {
        return true;
    }
    else {
        str++;
        return containsCharacter(target, str);
    }
}
int main() {
    char charArray[10] = { 'M','a','t','h','e','m','a','t','i','\0' };
    char targetCharacter = 'i';
    bool found = containsCharacter(targetCharacter, charArray);
    cout << "Contains character '" << targetCharacter
        << "': " << (found ? "true" : "false") << endl;
    return 0;
}
```

Output:
i: 1

3)



```
int recursiveFunction(int x, int y) {
    if (x == 0) {
        return y;
    }
    else {
        return recursiveFunction(x - 1, x + y);
    }
}

int main() {
    int result = recursiveFunction(5, 5);
    std::cout << "Result: " << result << std::endl;
    return 0;
}
```

Output:
20

4)

```
int Check(int n){
        if (n<=0)
                return 1;
        else
                return n + Check(n / 10);
}
int main(){
        cout<<Check(222);
        return 0;
}
```

Output:
247

Question 2: Implement a C++ function to find the middle node in a singly-linked list using recursion. The function should take the head of the linked list as input and return a pointer to the middle node. If the list has an even number of nodes, return the second middle node. (10 marks).

```
class Node {
    int data;
    Node* next;
};
// Function to find middle node using recursion
ListNode* findMiddleOfLinkedList(ListNode* slow, ListNode* fast) {

    if (fast == nullptr || fast->next == nullptr) {
        return slow;
    }
    else {

        return findMiddleOfLinkedList(slow->next, fast->next->next);
    }

}
int main() {

    Node* head = NULL;
    //Consider nodes are already inserted in the linked list



    ListNode* middle = findMiddleOfLinkedList(head, head);
    if (middle != NULL) {
        cout << "The middle node is: " << middle->data << endl;
    }
    else {
        cout << "The list is empty." << endl;
    }
  return 0;
}
```