|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **National University of Computer and Emerging Sciences, Lahore Campus** | | | | |
|  | **Program:** | **BS (Computer Science)** | **Semester:** | **Spring 2020** |
| **Duration:** | **60 Minutes** | **Total Marks:** | **25** |
| **Paper Date:** | **26th Feb-2020** | **Page(s):** | **3** |
| **Section:** | **All sections** | **Roll No.:** |  |
| **Exam:** | **MIDTERM-1** |  |  |
| **Instruction/Notes:** | * Understanding question is a part of your exam. In case of ambiguity, write your assumptions and answer accordingly * Time management is the key to success | | | |

**Q1: [5]**

Calculate time complexity of the following piece of code by formulating 𝑇(𝑛) and computing Big-O (tight bound).

|  |  |
| --- | --- |
| **int x=0;**  **for (int i=0; i< N; ++i){**  **for (int j=1; j<= 100; ++j){**  **for (int k=j; k<N; ++k){**  **x=x+i+j+k;**  **}**  **}**  **}** |  |
|  |
|  |
|  |
|  |
|  |
|  |

**Q2: [5]**

Write down code/ pseudocode using stack to determine if an input character string is of the form

Where, A and B are substrings in the reverse order. For example, if *A = abc*, and *B = cba* , then abc ∂ cba must return **true**. Else, it must return false.

#include <malloc.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char\* stack;

int top = -1;

// push function

void push(char ele)

{

stack[++top] = ele;

}

// pop function

char pop()

{

return stack[top--];

}

// Function that returns 1

// if str is a palindrome

int isPattern(char str[])

{

int length = strlen(str);

// Allocating the memory for the stack

stack = (char\*)malloc(length \* sizeof(char));

// Finding the mid

int i, mid = length / 2;

for (i = 0; i < mid; i++) {

push(str[i]);

}

// Checking if the length of the string

// is odd, if odd then neglect the

// middle character

if (length % 2 != 0) {

i++;

}

// While not the end of the string

while (str[i] != '\0') {

char ele = pop();

// If the characters differ then the

// given string is not a palindrome

if (ele != str[i])

return 0;

i++;

}

return 1;

}

// Driver code

int main()

{

char str[] = "madam";

if (isPattern(str)) {

cout <<"Yes";

}

else {

cout << "No";

}

return 0;

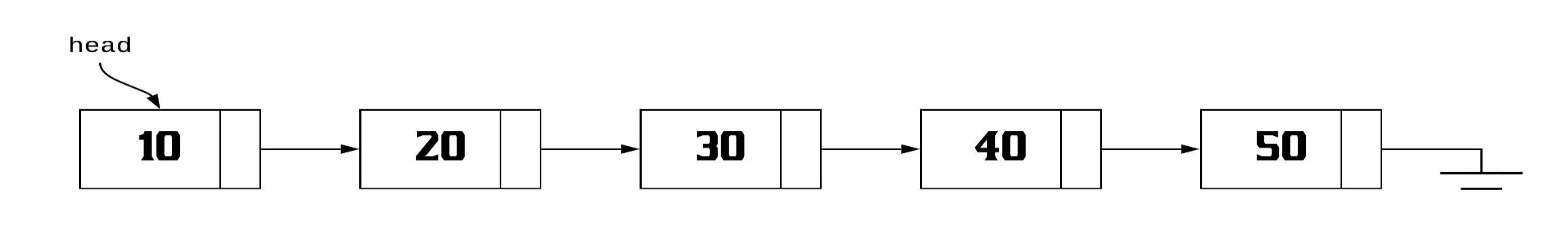
}

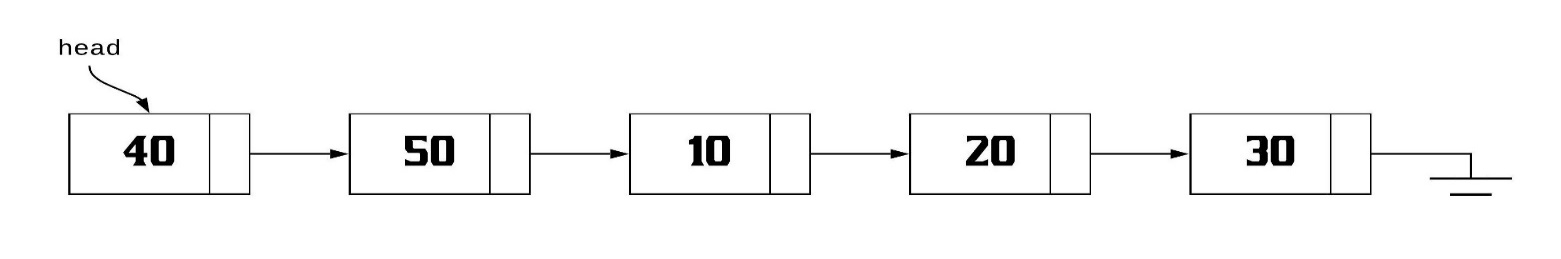
**Q3:**  **[10]**

Given the following code for a Linked List:

|  |
| --- |
| **class LinkedList {**  **struct Node {**  **int data;**  **Node\* next;**  **};**  **Node\* head;**  **public:**  **LinkedList() {**  **head = 0;**  **}**  **bool changeHead(int pos) {**  **//Your implementation**  **}**  **};** |

Provide an implementation of the function **changeHead** which changes the head of the Linked List to the required position. After changing the head, the list remains a valid Linked List.

A graphical representation of a Linked List is given below:

After calling **changeHead(3)** on this list, the resultant Linked List will look something like this:****

bool changeHead(int pos) {

if (pos < 0) {

return false;

}

if (pos == 0) {

return true;

}

Node\* curr = head;

int i = 0;

while (curr && i != pos - 1) {

curr = curr->next;

++i;

}

if (!curr)

return false;

Node\* temp = curr;

while (curr->next) {

curr = curr->next;

}

curr->next = head;

head = temp->next;

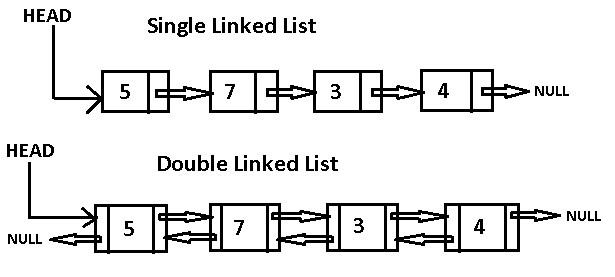
temp->next = nullptr;

return true;

}

**Q4: [5]**

Consider the following function that takes reference to head of a Doubly Linked List as parameter. Assume that a node of doubly linked list has previous pointer as *prev* and next pointer as *next.* Suppose that reference of head of following Doubly Linked List is passed as an argument to the function



**What will be the output of the function? Draw the modified linked list after the function call?**

**void foo(struct node \*&head\_ref)**

**{**

**struct node \*temp = NULL;**

**struct node \*current = head\_ref;**

**while (current !=  NULL)**

**{**

**temp = current->prev;**

**current->prev = current->next;**

**current->next = temp;**

**current = current->prev;**

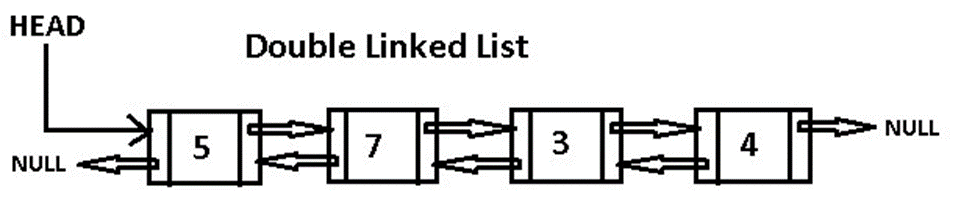
**}**

**if(temp != NULL )**

**head\_ref = temp->prev;**

Solution:

The given function reverses the doubly linked list.



**4**

**5**

**7**

**3**