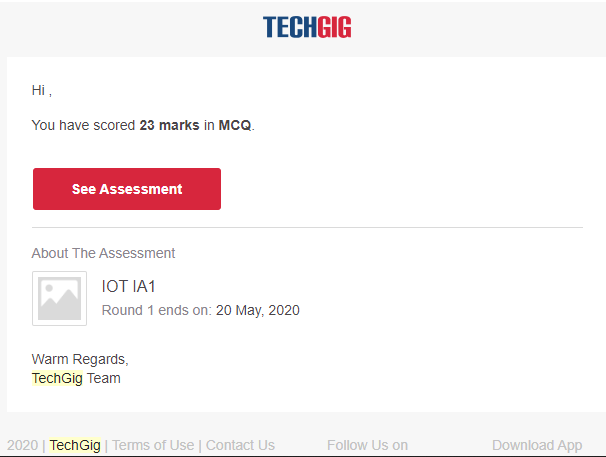
**DAILY ONLINE ACTIVITIES SUMMARY**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **20/05/2020** | | | | | **Name:** | **Manish B Shriyan** | |
| **Sem & Sec** | **8th sem B sec** | | | | | **USN:** | **4AL16CS131** | |
| **Online Test Summary** | | | | | | | | |
| **Subject** | | **IOT** | | | | | | |
| **Max. Marks** | | **30** | | **Score** | | | **23** | |
| **Certification Course Summary** | | | | | | | | |
| **Course** | **AWS Cloud Practitioner Essentials(Second Edition): AWS Integrated Services** | | | | | | | |
| **Certificate Provider** | | | **AWS** | | **Duration** | | | **100 Mins** |
| **Coding Challenges** | | | | | | | | |
| **Problem Statement:**   1. **If a linked listis: 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8**   **The value of size k is 2**  **Then the linked list looks like: 2 → 1 → 4 → 3 → 6 → 5 → 8 → 7**   1. **If a linked listis: 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8**   **The value of size k is 3**  **Then the linked list looks like: 3 → 2 → 1 → 6 → 5 → 4 → 8 → 7** | | | | | | | | |
| **Status: Solved** | | | | | | | | |
| **Uploaded the report in Github** | | | | | **Uploaded** | | | |
| **If yes Repository name** | | | | | **ManishShriyan** | | | |
| **Uploaded the report in slack** | | | | | **Yes** | | | |

Online Test Details:



Certification Course Details:



Coding Challenges Details:

|  |
| --- |
| 1: |
|  | If a linked listis: 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 |
|  | The value of size k is 2 |
|  | Then the linked list looks like: 2 → 1 → 4 → 3 → 6 → 5 → 8 → 7 |
|  |  |
|  | Test Case 2: |
|  | If a linked listis: 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 |
|  | The value of size k is 3 |
|  | Then the linked list looks like: 3 → 2 → 1 → 6 → 5 → 4 → 8 → 7 |
|  |  |
|  | strustruct Node |
|  |  |
|  |  |
|  | { |
|  | int data; |
|  | struct Node\* next; |
|  | }; |
|  | pointer to the new head node. / |
|  | struct Node reverse (struct Node head, int k) |
|  | { |
|  | struct Node current = head; |
|  | struct Node next = NULL; |
|  | struct Node prev = NULL; |
|  | int count = 0; |
|  | while (current != NULL && count < k) { next = current->next; current->next = prev; prev = current; current = next; count++; } if (next != NULL) head->next = reverse(next, k); return prev; |
|  | } |
|  | void push(struct Node\*\* head\_ref, int new\_data) |
|  | { |
|  | struct Node\* new\_node = |
|  | (struct Node\*) malloc(sizeof(struct Node)); |
|  | new\_node->data = new\_data; new\_node->next = (\*head\_ref); (\*head\_ref) = new\_node; |
|  | } |
|  | void printList(struct Node node) |
|  | { |
|  | while (node != NULL) |
|  | { |
|  | printf("%d ", node->data); |
|  | node = node->next; |
|  | } |
|  | } |
|  | int main(void) |
|  | { |
|  | struct Node head = NULL; |
|  | push(&head, 8); |
|  | push(&head, 7); |
|  | push(&head, 6); |
|  | push(&head, 5); |
|  | push(&head, 4); |
|  | push(&head, 3); |
|  | push(&head, 2); |
|  | push(&head, 1); |
|  | printf("\nGiven linked list \n"); printList(head); head = reverse(head, 2); printf("\nReversed Linked list \n"); printList(head); |
|  | return(0); |
|  |  |
|  |  |
|  | ================================================================================================================================== |
|  | ================================================================================================================================== |
|  |  |
|  |  |
|  | ONLINE CODING(20/5/2020) |
|  |  |
|  | Write a C Program to Reverse a Linked List in groups of given size. |
|  | #include<stdio.h> |
|  | #include<stdlib.h> |
|  |  |
|  | struct Node |
|  | { |
|  | int data; |
|  | struct Node\* next; |
|  | }; |
|  | pointer to the new head node. / |
|  | struct Node reverse (struct Node head, int k) |
|  | { |
|  | struct Node current = head; |
|  | struct Node next = NULL; |
|  | struct Node prev = NULL; |
|  | int count = 0; |
|  | while (current != NULL && count < k) |
|  | { |
|  | next = current->next; |
|  | current->next = prev; |
|  | prev = current; |
|  | current = next; |
|  | count++; |
|  | } |
|  | if (next != NULL) |
|  | head->next = reverse(next, k); |
|  | return prev; |
|  | } |
|  | void push(struct Node\*\* head\_ref, int new\_data) |
|  | { |
|  | struct Node\* new\_node = |
|  | (struct Node\*) malloc(sizeof(struct Node)); |
|  | new\_node->data = new\_data; |
|  | new\_node->next = (\*head\_ref); |
|  | (\*head\_ref) = new\_node; |
|  | } |
|  | void printList(struct Node node) |
|  | { |
|  | while (node != NULL) |
|  | { |
|  | printf("%d ", node->data); |
|  | node = node->next; |
|  | } |
|  | } |
|  | int main(void) |
|  | { |
|  | struct Node head = NULL; |
|  | push(&head, 8); |
|  | push(&head, 7); |
|  | push(&head, 6); |
|  | push(&head, 5); |
|  | push(&head, 4); |
|  | push(&head, 3); |
|  | push(&head, 2); |
|  | push(&head, 1); |
|  | printf("\nGiven linked list \n"); |
|  | printList(head); |
|  | head = reverse(head, 2); |
|  | printf("\nReversed Linked list \n"); |
|  | printList(head); |
|  | return(0); |