CSCI 2270 - Zagrodzki, Ashraf, Trivedi - CS2: Data Structures

Started on	Sunday, 9 February 2020, 5:50 PM
State	Finished
Completed on	Sunday, 9 February 2020, 6:17 PM
Time taken	26 mins 21 secs
Marks	8.00/9.00
Grade	**8.89** out of 10.00 (**89**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Linked list is generally considered as an example of _____ type of memory allocation.

Select one:

- a. static
- b. dynamic ✔
- c. fixed
- d. All of the above
- e. None of the above

Your answer is correct.

The correct answer is: dynamic

Question **2**

Correct

Mark 1.00 out of 1.00

While traversing a singly linked list, we access each node using:

Select one:

- a. pointers ✔
- b. index
- c. reference
- d. node data
- e. None of the above

Your answer is correct.

The correct answer is: pointers

Question **3**

Correct

Mark 1.00 out of 1.00

Which of the following statements is *not* a feature of linked list?

Select one:

- Random access — for example, accessing element 5, then 17, and then 9 — is efficient ✔
- Take up memory that is linear to the length of list
- Do not need to move elements when deleting or inserting
- Do not need to preestimate the memory required

Your answer is correct.

Linked list cannot randomly access to elements because the nodes are not stored in a contiguous area of memory. It has to iterate through the begining until the targets thus is not efficient.

Linked list cannot randomly access to elements because the nodes are not stored in a contiguous area of memory. It has to iterate through the begining until the targets thus is not efficient.

The correct answer is: Random access — for example, accessing element 5, then 17, and then 9 — is efficient

Question **4**

Incorrect

Mark 0.00 out of 1.00

Which of the following statements about linked list is **correct**?

Select one:

- We need to delete the memory of a removed node in order to avoid memory leak
- *Head* must point to a node.
- Deleting memory pointed by the head deletes the whole list. ✖
- All of the nodes in a linked list must be stored in a contiguous area of memory.

Your answer is incorrect.

The correct answer is: We need to delete the memory of a removed node in order to avoid memory leak

Question **5**

Correct

Mark 1.00 out of 1.00

Which of the following can tell us the linked list is empty?

Select one:

- if(head==NULL) ✔
- if(head)
- if(head->key == 0)
- if(head->next==NULL)

Your answer is correct.

The correct answer is: if(head==NULL)

Question **6**

Correct

Mark 1.00 out of 1.00

Consider a linked list implementation where we have both a pointer to the head of the list (called head) and a pointer to the tail of the list (called tail).

For this new data structure, which of the following operations will require traversing the whole list?

Select one:

- a. Delete the first node
- b. Insert a new node after the last node
- c. Insert a new node before the first node
- d. Delete the last node ✔ For deleting the last node, refer to the recitation write up, we need two pointers to keep track of the previous node that points to the last node, so inevitably we need loop through all the linked list so the length matters.

Your answer is correct.

Obviously the operations related to the first element has nothing to do with the list's length.

If we have such a *tail* to be the reference of last node, when we want to add a new node after the last node, the only thing to do is to
1. locate the last node by this tail
2. update the next pointer of the current last node to the new node
3. make the new node point to NULL and make *tail* to be the reference of the new node.

So adding a new node after the last one does not need to loop through all the list.

For deleting the last node, refer to the recitation write up, we need two pointers(*prev* and *pres*) to keep track of the previous node that points to the last node, so inevitably we need loop through all the linked list so the length matters.

The correct answer is: Delete the last node

Question **7**

Correct

Mark 1.00 out of 1.00

Consider the following structure of node and linked list.

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

-1 -> 5 -> 30 -> 5 -> 5 -> 15 -> 5 -> 17 NULL

What will be the output of following pseudo-code? Consider head is the pointer to the first node of above linked list.

```
Node *pres = head;
int count = 0;
while(pres!= NULL && count < 3) {
        if(pres->key == 5) {
            count = count + 1;
        }
        pres = pres->next;
}
cout <<  pres->key  << endl;
```

Select one:

- a. 30
- b. 5
- c. 15 ✔
- d. 17
- e. None of above

Your answer is correct.

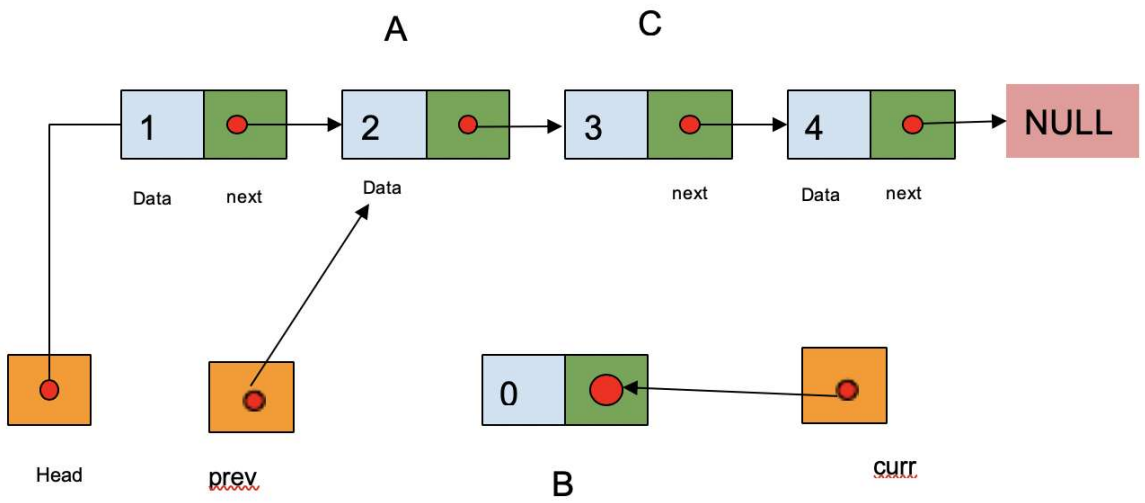
The correct answer is: 15

Question **8**

Correct

Mark 1.00 out of 1.00

Given a linked list below, we would like to insert a new node **B** between **A** and **C**, which of the following code segment is correct? Notice that pointer *prev* points to node **A** and pointer *curr* points to node **B.**



Select one:

- **prev -> next = curr;**
  curr -> next = prev -> next;
- **curr -> next = prev -> next;** ✔
  prev -> next = curr;
- **curr -> next = prev -> next;**
  prev -> next = curr -> next -> next;
- **curr = prev;**
  prev = curr->next;

Your answer is correct.

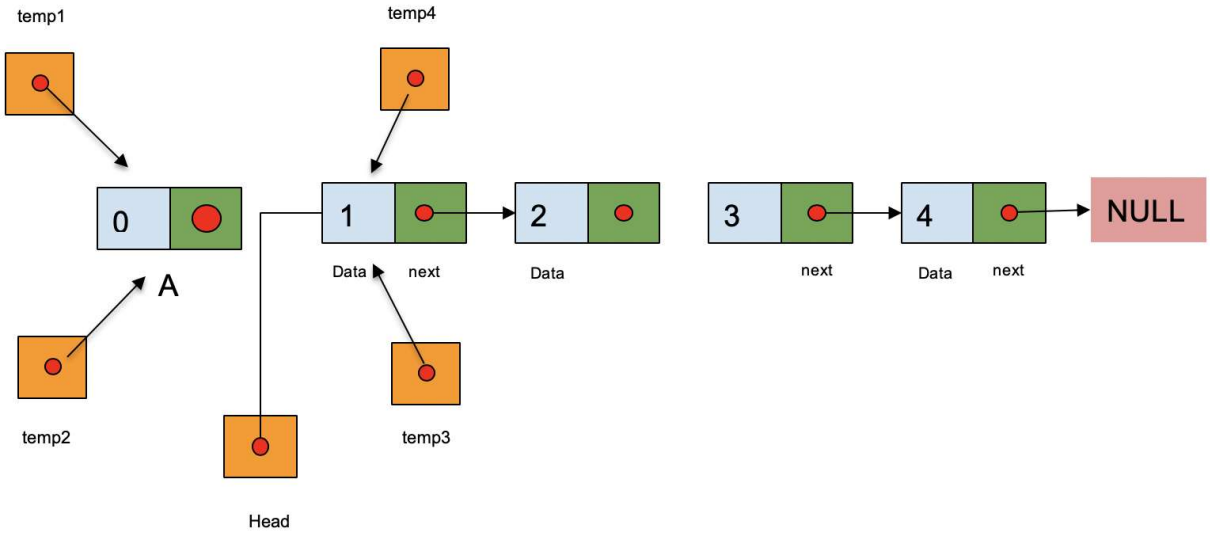
The correct answer is: **curr -> next = prev -> next;**
prev -> next = curr;

Question **9**

Correct

Mark 1.00 out of 1.00

Given a linked list below, we would like to insert a new node **A** at the start of the list. Which of the following code segments are correct? Select all that applies. Notice that pointer *temp1* and *temp2* point to node **A**, pointer *temp3* and *temp4* point to the original first node.



Select one or more:

- [x] **temp1 -> next = head;**
  head = temp1; ✔
- [x] **head = temp1;**
  head -> next = temp3 ; ✔
- [ ] **head = head->next;**
  temp2->next = head;
- [ ] **temp1->next = temp4;**
  temp2 -> next = temp3;
  temp1 = temp2;

Your answer is correct.

The correct answers are: **head = temp1;**
head -> next = temp3 ;, **temp1 -> next = head;**
head = temp1;

Question **10**

Complete

Not graded

```
1   #include<iostream>
2   int main(int argc, char* argv[])
3   {
4       int *pa = new int[10];
5       int *pb = new int[20];
6       int *tmp = NULL;
7
8       tmp = pb;
9       pb = pa;
10
11      delete []pa;
12      delete [] pb;
13      delete []tmp;
14
15      return 0;
16  }
```

Which of the following options are true? Select all that apply.

Select one:

- Memory block allocated by Line 4 is not released.
- Memory block allocated by Line 5 is not released.
- Pointer in Line 6 can't be assigned to NULL.
- Memory block allocated by Line 4 is released more than once.
- In line 12, space between [ ] and pb is not allowed.

Your answer is partially correct.

The correct answers are: Memory block allocated by Line 4 is not released., Memory block allocated by Line 5 is not released., Pointer in Line 6 can't be assigned to NULL. , In line 12, space between [ ] and pb is not allowed., Memory block allocated by Line 4 is released more than once.