Quizizz

NAME : ______________________

CLASS : ______________________

DATE : ______________________

DS UNIT-2 TEST-1

10 Questions

1. What does the following function do for a given Linked List with first node as *head*?

```
void fun1(struct node* head) {
if(head == NULL)
return;
fun1(head->next);
printf("%d ", head->data); }
```

A Prints alternate nodes of Linked List

B Prints all nodes of linked lists

C Prints alternate nodes in reverse order

D Prints all nodes of linked list in reverse order

2. Which of the following points is/are true about Linked List data structure when it is compared with array?

A. The size of array has to be pre-decided, linked lists can change their size any time

B. It is easy to insert and delete elements in Linked List

C. Random access is not allowed in a typical implementation of Linked Lists

D. ALL THE ABOVE

E. Arrays have better cache locality that can make them better in terms of performance

3. 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*.

```
void fun(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; }
```

Assume that reference of head of following doubly linked list is passed to above function
1 <--> 2 <--> 3 <--> 4 <--> 5 <-->6.
What should be the modified linked list after the function call?

A	2 <--> 1 <--> 4 <--> 3 <--> 6 <-->5	B	5 <--> 4 <--> 3 <--> 2 <--> 1 <-->6.
C	6 <--> 5 <--> 4 <--> 3 <--> 2 <--> 1	D	6 <--> 5 <--> 4 <--> 3 <--> 1 <--> 2

4. Which of the following sorting algorithms can be used to sort a random linked list with minimum time complexity?

A	Merge Sort	B	Selection Sort
C	Insertion Sort	D	Quick Sort

5. The following function reverse() is supposed to reverse a singly linked list. There is one line missing at the end of the function.

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

/* head_ref is a double pointer which points to head (or start) pointer
  of linked list */
static void reverse(struct node** head_ref) {
    struct node* prev   = NULL;
    struct node* current = *head_ref;
    struct node* next;
    while (current != NULL)     {
        next  = current->next;
        current->next = prev;
        prev = current;
        current = next;     }
    /*ADD A STATEMENT HERE*/
}
```

What should be added in place of "/*ADD A STATEMENT HERE*/", so that the function correctly reverses a linked list.

A *head_ref = next;

B *head_ref = NULL;

C *head_ref = prev;

D *head_ref = current;

6. What is the output of following function in which start is pointing to the first node of the following linked list 1->2->3->4->5->6 ?

```
void fun(struct node* start) {
  if(start == NULL)
    return;
  printf("%d  ", start->data);
  if(start->next != NULL )
    fun(start->next->next);
  printf("%d  ", start->data); }
```

A 1 2 3 5

B 1 4 6 6 4 1

C 1 3 5 1 3 5

D 1 3 5 5 3 1

7. The following C function takes a simply-linked list as input argument. It modifies the list by moving the last element to the front of the list and returns the modified list. Some part of the code is left blank. Choose the correct alternative to replace the blank line.

```
typedef struct node {
  int value;
  struct node *next; }Node;

Node move_to_front(Node head) {
  Node p, q;
  if ((head == NULL: || (head->next == NULL))
    return head;
  q = NULL; p = head;
  while (p-> next !=NULL)   {
    q = p;
    p = p->next;
  }
 //FILL THE MISSING CODE HERE //
  return head;
}
```

A
```
q = NULL;
p->next = head;
head = p;
```

B
```
q->next = NULL;
head = p;
p->next = head;
```

C
```
head = p;
p->next = q;
q->next = NULL;
```

D
```
q->next = NULL;
p->next = head;
head = p;
```

8. In the worst case, the number of comparisons needed to search a singly linked list of length n for a given element is

A n

B log(2*n)

C n/2

D log(2*n)-1

9. Suppose each set is represented as a linked list with elements in arbitrary order. Which of the operations among union, intersection, membership, cardinality will be the slowest?

A cardinality

B union, intersection

C membership

D union

10. Consider the function f defined below.

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

int f(struct item *p) {
  return (
    (p == NULL) ||          (p->next == NULL) ||    (( p->data <= p->next->data) && f(p->next))
 );
}
```

A not all elements in the list have the same data value.

B the elements in the list are sorted in non-increasing order of data value

C None of the Above

D the elements in the list are sorted in non-decreasing order of data value

Answer Key

1. d	2. d	3. c	4. a
5. c	6. d	7. d	8. a
9. b	10. d		