**Multiple Choice Questions**

1.Option D

2.Option B

3.Option B

4.Option B

5.Option A

6. Option D

7. Option B

8.2000

9. Option A #

10. Option A

11. Option B

12. Option C

13. Option A

14. Option A

15. Option A

17. Option B

18. Option C

**Programming Questions**

1. #include<stdio.h>

typedef struct node{

int data;

struct node \*next;

}node;

node\* shuffle(struct node\* head)

{

node slow = head,fast = head,first=head,second,first\_next,second\_next;

//Find offset of the middle element using fast pointer method

while(!fast && !fast-next){

slow = slow->next;

fast = fast->next->next;

}

second=slow;

new\_head = second;

while(second)

{

first\_next = first->next;

second\_next = second->next;

second\_next = first;

first->next = second;

first = first\_next;

second = second\_next;

}

return new\_head;

}

Time:O(n) Space:O(1)

2. #include<stdio.h>

void ways(int total,int k)

{

/\*

If n is 0 do nothing is the base case.

There are 2 choices . Either subtract number in range[1..n] from n or not . For the 2 cases we recursively print the combination.

\*/

int i,j,m;

if (!(total && k))

printf("\n");

m = total-k;

i=total;

j=k;

if(m>=0 && i>j){

ways(total,k-1);

printf("%d +",k);

ways(total-k,k);}

else if(i<j)

ways(total,k-1);

else if(m<j){

ways(total,k-1);

printf("%d + ",k);

ways(k,k);}

}

int main(int argc, char const \*argv[])

{

ways(4,4);

return 0;

}

Time:O(2^n) Space:O(n)