1.

```
#include <stdio.h>
#include <stdio.h>
void ins (node*, int, int)
Int size = 0,
Struct nodes
Int data;
Struct node* next,
};
Node* get node (int data)
{
Node* new node = (instruct node*) malloc
                 (new node);
new node -> data = data;
new node -> next = null;
return new node;
}
void ins(node* current, int pos, int data)
{
if (pos <- 1 || pos > size+1)
    printf ("Invalid ");
else
```

```
{
while (pos--)
{
if (pos==0)
{
node *temp=getnode(data);
temp->next = *current;
*current=temp;
}
else
{
current=&(*current)->next;
}
size++;
}
}
void print(struct node *head)
{
while(h!=NULL)
{
printf("%d", head->data);
head = head->next;
}
printf("\n");
}
void del(Struct node* head_ref, int pos)
{if (head_ref==NULL)
return;
temp= head_ref;
if (pos=0)
{
```

```
    * head_ref = temp -> next;
      free (temp);
      return;
    for (int i=0; temp != NULL && i<pos-1; i++)
        temp = temp->next;
        free (temp -> next);
        temp -> next = next;

    }
    int main ()
    {
     struct node * head = NULL;
     push (&head, 7);
     push (&head, 8);
     push (&head, 6);
     ins (&head, 7, 5);
     del (&head, 4);
     print list (head);
     return (0);

    }
```

```
#include <stdio.h>
#include <stdio.h>
struct node
{
int data;
struct node * next;
};
void print list (struct node * head)
{
struct node * ptr = head;
while (ptr)
{printf ("%d →", ptr -> data);
ptr = ptr → next;
}
Print ("NULL] n");
}
void push (struct node* head, int data)
{
struct Node* new = (struct node*) malbe (size
of (struct node));
new → data = data;
new → next = *head;
* head = new;
}
struct node* merge (struct node* a, struct node* b)
{
struct node during;
struct node * fail = dumy;
dumy next = NULL;
while (1)
{
```

```
if (a == NULL)
{ tail -> next = b;
  break;
}
else if (b == NULL)
{
  tail -> next = a;
  break;
}
else
{ tail -> next = a;
  tail = a;
  a = a -> next;
  tail -> next = b
  tail = b;
   b = b -> next;
}
}
return dummy next;
}
void main()
{
int keys[] = {1,2,3,4,5,6,7};
int n = sizeof keys) (sizeof key[0];
Struct product a = Null , b = Null;
for ( int i=n-1 ; i>0 ; i = i-2)
    Push (&a, keys[i]);
for ( int i=n-2; i>=0 ; i = i-2)
  Push (&b, Key(i)};
```

```
struct node *head = merge(a, b);
print list (head);
}
```

3)
```
#include<stdio.h>
void find (int ar[], int n, int s)
{
int sum=0;
int i=0;i<n;i++)
{
while (sum<s && i<n)
{
sum+=ar[i];
i++;
```

```
if (sum == s)
{
printf ("found");
return;
}
sum -= arr[b];
}
}
int main (void)
{
int arr[] = {2,5,0,9,7,3}
int s=15;
int n= size of (arr) / size of (arr[0]);
find (arr, n, s);
return 0;
}
```

4.
```
#include <stdio.h>
#include <stdio.h>
struct node
{
int data;
struct node * next;
};
void printer (struct node * head)
{
if (head == NULL)
    return;
printrer (head -> next);
print f ("%d", head -> data);
void push (struct node* head yet, char
                                    new)
{
struct node * node new = (struct node*)
  malloc (size of (struct node));
  node-> new-> data = new;
  node- new -> rent = (*head-ref);
  (head_ref) = node -new;
}
int main ()
struct node* head = Null;
Push (& head, 4);
Push (& head, 3);
Push ( & head, 2);
Print new (head); Print a normale (head);
return 0; }
```

```
void print alternate (struct node* head)
{
int count = 0;
while (head != NULL)
  { if (count % 2 == 0)
     cout << head -> data << " ";
     count ++;
     head = head -> next
  }
```

5) i. Array is a set of similar objects stored in sequential memory allocation under variable name but linked list is a data structure which contains a sequence of the elements where each element is linked to its next element.

ii. #include <stdio.h>

```
int main()
{
int a1[100], a2[100];
int n, x, pos, n=10;
for (i=0; i<10; i++)
  scanf("%d", &a1[i]); scanf("%d", &a2[i]);
for (i=0; i<n; i++)
    printf("%d", a[1])
    printf("%d", a[2])

x = a2(0);
pos = 1;
n++;
for (i=n; i>=pos; i--)
    a1[i] = a1[i-1];
```

```
a[pos-1] = x;
for(i=0; i<n; i++):
    printf("%d", a1[i]);
for(i=1; i<n; i++)
    Printf("%d", a2[i]);
return 0;
}
```