```c
// /*
#include<stdio.h>
#include<stdlib.h>
// ***************************
struct node
{
 int data;
 struct node*next;
};
//*****************************
struct node*createnode()             //node creation
{
 int data;
 struct node*newnode = NULL;
 newnode = (struct node*)malloc(sizeof(struct node));
 if(newnode == NULL)
 {
 printf("Memory not allocated\n");
 return NULL;
 }
 else
 {
  printf("Enter the data\n");
  scanf("%d",&data);
  newnode->data = data;
  newnode->next = NULL;
  return newnode;
 }
}
//******************************
void create_linklist(struct node**head)          //createlinklist
{
  struct node*newnode = NULL;
  struct node*travnode = *head;
  newnode = createnode();
  if(*head == NULL)
  {
   *head = newnode;
  }
  else
  {
  while(travnode->next!=NULL)
  {
   travnode = travnode->next;
  }
   travnode->next = newnode;
 }
}
//*******************************
int countnode(struct node*head)        //countnode;
{
  int count = 0;
  while(head!=NULL)
  {
   count++;
   head = head->next;
  }
  return count;
}
//*******************************
void display_linklist(struct node*head)
{                                                  //display linklist
 printf("Your linklist is \n");
 while(head!=NULL)
 {
  printf("%d\t",head->data);
  head = head->next;
 }
 printf("\n");
}
//********************************
void insert_at_first(struct node**head)           //insert at first position
{
 struct node*newnode = NULL;
 newnode = createnode();
 newnode->next = *head;
 *head = newnode;
}
//*****************************
void insert_at_position(struct node**head)       //insert at a position
{
 int pos,count;
 printf("Enter the position at which you want to insert a node\n");
 scanf("%d",&pos);
 count = countnode(*head);
 if(pos == 1)
 {
   insert_at_first(head);
 }else if(pos == count+1)
         {
           create_linklist(head);
         }
  else if(pos < 1 || pos > count+1)
         {
          printf("Invalid position to insert a node\n");
          insert_at_position(head);
         }
  else
  {
   struct node *newnode = NULL;
   newnode = createnode();
   struct node *tempnode = *head;
   for(int i=1; i < (pos-1); i++)
      {
        tempnode = tempnode->next;
      }
    newnode->next = tempnode->next;
    tempnode->next = newnode;
  }
}
//********************************
void delete_first(struct node**head)      //delete first node
{
  struct node * ptr = *head;
  *head = ptr->next;
  free(ptr);
}
//*********************************
void delete_last(struct node **head)
{
 struct node *tempnode = *head;                    //delete last node
 while(tempnode->next->next != NULL)
 {
  tempnode = tempnode->next;
 }
 struct node *ptr = tempnode->next;
 tempnode->next = NULL;
 free(ptr);
}
//*************************************
void delete_at_position(struct node**head)
{
 int pos,count;                                                //delete at
position
 printf("Enter a position at which you want to delete a node\n");
 scanf("%d",&pos);
 count = countnode(*head);
 if(pos == 1)
 {
  delete_first(head);
 }
 else if(pos < 1 || pos > count)
      {
       printf("Invalid position to delete a node\n");
       delete_at_position(head);
      }
 else if(pos == count)
      {
       delete_last(head);
      }
 else
 {
   struct node*tempnode =*head;
   for(int i=1;i<count-1;i++)
    {
      tempnode = tempnode->next;
    }
   struct node* ptr = tempnode->next;
   tempnode->next = ptr->next;
   free(ptr);
  }
}
//*****************************************
void reverse_linklist(struct node**head)    //Reverse singly_linklist
{
 //int temp;
 struct node * travnode = *head;
 if(*head == NULL)
 {
  printf("LINKLIST DOSENT EXIT !!!!!!!\n");
 }
 struct node *temp = NULL;
 struct node * prev = NULL;
 while(travnode!= NULL)
   {
   temp = travnode->next;
   travnode->next = prev;
   prev = travnode;
   travnode = temp;
   }
 *head = prev;
 }
//******************************************
void delete_at_data(struct node**head)               //delete at given data
{
 int data,flag;
 printf("Enter a data of node which you want to delete\n");
 scanf("%d",&data);
 struct node*travnode = *head;
 //struct node*tempnode = NULL;
 int first = travnode->data;
  if(data == first)
  {
    delete_first(head);
  }else

 while(travnode->next!=NULL)
 { flag = 0;
```

```
   if(travnode->next->data == data && travnode->next->next == NULL)
  {
     delete_last(head);
     flag =1;
     break;
  }
  else if(travnode->next->data== data)
  {
  struct node* tempnode = travnode->next;
   travnode->next = tempnode->next;
   free(tempnode);
   flag = 1;
   break;
  }
  travnode = travnode->next;
 }
 if(flag == 0)
  {
   printf("data not in list\n");
   delete_at_data(head);
 }
 }

//************************************************
void sort_linklist(struct node**head)                    // sorting
{
 int swap;
 int count = countnode(*head);
 struct node*travnode = *head;
 for(int i =0;i<count-1;i++)
  {
   for(int j=0;j<count-i-1;j++)
    {

      if(travnode->data >travnode->next->data)
       {
        swap = travnode->next->data;
        travnode->next->data = travnode->data;
        travnode->data = swap;

       }
    travnode= travnode->next;
   }
   travnode = *head;
 }
}
//**************************************************
void print_at_evenpos(struct node**head)                    //print data @even
position
{
  int count = countnode(*head);
  struct node*travnode = *head;
  for(int i = 1;i<count+1;i++)
   {
     if(i%2 == 0)
      {
        printf("%d\t",travnode->data);
      }

      travnode = travnode->next;
  }printf("\n");
}
//****************************************************
void main()
{
 struct node *first = NULL;                             //main function
 int choice;
 do
 {
  printf("1. Create linklist\n");
  printf("2. Display linklist\n");
  printf("3. Insert at first\n");
  printf("4. Insert at a position\n");
  printf("5. Delete first node\n");
  printf("6. Delete node at a position\n");
  printf("7. Delete last node\n");
  printf("8. Reverse the linklist\n");
  printf("9. Delete at given data\n");
  printf("10. Sort the linklist\n");
  printf("11. Print data on even position\n");
  printf("12 Exit\n");
  printf("******************************\n");
  printf("Enter your choice\n");
  scanf("%d",&choice);
  switch(choice)
  {
   case 1: create_linklist(&first);
           break;
   case 2: display_linklist(first);
           break;
   case 3: insert_at_first(&first);
           break;
   case 4: insert_at_position(&first);
           break;
   case 5: delete_first(&first);
           break;
   case 6: delete_at_position(&first);
           break;
   case 7: delete_last(&first);
           break;
   case 8: reverse_linklist(&first);
           break;
   case 9: delete_at_data(&first);
           break;
   case 10: sort_linklist(&first);
            break;
   case 11: print_at_evenpos(&first);
            break;
  }
 }while(choice!=12);
}
  //*/
  //************************************************************************

    //LINKED LIST OF PRIME NUMBERES BETWEEN 1 TO 1000

 // /*
#include<stdio.h>
#include<stdlib.h>
// ***************************
struct node
{
 int data;
 struct node*next;
};
//****************************
struct node*createnode(int a)              //node creation
{
 int data = a;
 struct node*newnode = NULL;
 newnode = (struct node*)malloc(sizeof(struct node));
 if(newnode == NULL)
 {
 printf("Memory not allocated\n");
 return NULL;
 }
 else
 {
  //printf("Enter the data\n");
  //scanf("%d",&data);
  newnode->data = data;
  newnode->next = NULL;
  return newnode;
 }
}
//******************************

void create_linklist(struct node**head,int a)          //createlinklist
{
 int pass = a;
  struct node*newnode = NULL;
  struct node*travnode = *head;
  newnode = createnode(pass);
  if(*head == NULL)
  {
   *head = newnode;
  }
  else
  {
  while(travnode->next!=NULL)
  {
   travnode = travnode->next;
  }
   travnode->next = newnode;
 }
}
//*******************************
void display_linklist(struct node*head)
{
 printf("Your linklist is \n");
 while(head!=NULL)
 {
  printf("%d\t",head->data);
  head = head->next;
 }
 printf("\n");
}
//*******************************
void main()
{
 struct node*first = NULL;
 int i =1;

 while(i<1000)
 {
  int a = i,flag =0;
  for(int i=1;i<a;i++)
  {
   if(i==1)
    {
     continue;
    }
   if(a%i==0)
          {
            flag =1;

           }
  }
```

```
   if(flag== 0)
   {

   create_linklist(&first,i);
   }
   i++;
  }
  display_linklist(first);

   }

 // */
```