//Son Nguyen - 103234103

//Question 1 Write a program that creates a linked list of 10 characters, then reverse the order of the list

//(Creating a Linked List , then Reversing Its Element)//

#include <stdio.h>

#include <stdlib.h>

struct node {

char data;

struct node \*next;

};

typedef struct node Node;

typedef Node \*nodePtr;

void printList (nodePtr currentPtr)

{

while(currentPtr!=NULL)

{

printf("%c->", currentPtr->data);

currentPtr = currentPtr->next;

}

}

// function to reverse the link list

nodePtr reverseList (nodePtr currentPtr)

{

nodePtr previousPtr = NULL , nextPtr;

while (currentPtr!=NULL)

{

nextPtr=currentPtr->next;

currentPtr->next=previousPtr;

previousPtr=currentPtr;

currentPtr=nextPtr;

}

return previousPtr;

}

int main()

{

nodePtr startPtr=NULL, newPtr , currentPtr;

char letter = 'A';

for(int i=0;i<10;i++)

{

newPtr = (Node\*)malloc(sizeof(Node));

newPtr->data = letter;

newPtr->next = NULL;

if (startPtr == NULL)

{

startPtr = newPtr;

currentPtr = newPtr;

}

else

{

currentPtr->next = newPtr;

currentPtr = newPtr;

}

letter++;

}

//Bringing the pointer to start of the list

currentPtr=startPtr;

printf("The created list is:\n");

printList(currentPtr);

printf("NULL\n\n");

//Reverse order

//printf ("The list in reverse is: \n");

currentPtr=reverseList(currentPtr);

printList(currentPtr);

printf("NULL");

return 0;

}}  
  
**Output**  
 Graphical user interface, text, application, email

Description automatically generated  
--------------------------------------------------------------------------------------------------------------------------------------  
// Son Nguyen - 103234103 //

// Son Nguyen - 103234103 //

// Question 2 - Write a function printListBackward that recursively outputs the items in a list in reverse order//

//Use your function in a test program that creates a sorted list of integers and print the list in reverse order//

//(Recursively Print A List Backward)

#include<stdio.h>

#include<stdlib.h>

struct node

{

char data;

struct node \*next;

};

typedef struct node Node;

typedef Node \*nodePtr;

void printList (nodePtr currentPtr)

{

while(currentPtr!=NULL)

{

printf("%c->",currentPtr->data);

currentPtr = currentPtr -> next;

}

}

void print\_rev\_rec(nodePtr currentPtr)

{

if (currentPtr==NULL)

return;

else

print\_rev\_rec(currentPtr->next);

//calling the print reverse fuction recursively

printf("%c->", currentPtr -> data);

}

int main ()

{

nodePtr startPtr=NULL,newPtr,currentPtr;

char letter = 'A';

for (int i= 0;i<10;i++)

{

newPtr=(Node\*)malloc(sizeof(Node));

newPtr -> data = letter;

newPtr -> next = NULL;

if (startPtr == NULL)

{

startPtr = newPtr;

currentPtr = newPtr;

}

else

{

currentPtr->next = newPtr;

currentPtr = newPtr;

}

letter++;

}

// Moving current pointer back to beginning of the list

currentPtr = startPtr;

printf("The created list is:\n");

printList(currentPtr);

printf("NULL \n \n");

//calling the recursively printing function

printf("THe recursively reversed list is:\n");

print\_rev\_rec(currentPtr);

printf("NULL \n");

return 0;

}**Output**   
Graphical user interface, text, application

Description automatically generated