Name: Kaushal Oza

SRN: 201900754

Roll no: 40

#include<stdio.h>

#include<conio.h> #include<stdlib.h> #include<string.h> typedef struct node n; struct node

{

int power; struct node \*next;

};

n \*head1 = NULL; n \*head2 = NULL; n \*head = NULL; char\* xor(char a[],char b[])

{ int n,i; n = strlen(b); char \*res; free(res); res = (char \*)malloc(sizeof(char)\*n); for(i=0;i<n;i++)

{ if(a[i] == b[i]) \*(res+i)= '0'; else \*(res+i)= '1';

} \*(res+i) = '\0'; return res; } char\* divi(char divident[],char divisor[])

{ int n,divslen,i,s; divslen = strlen(divisor); s = strlen(divisor); char \*temp1,\*temp2; free(temp1); free(temp2); temp1 = (char \*)malloc(sizeof(char)\*s); temp2 = (char \*)malloc(sizeof(char)\*s); n = strlen(divident); for(i=0;i<divslen;i++) temp1[i] = divident[i]; temp1[i] = '\0'; while (divslen < n)

{ if (temp1[0] == '1')

{

strcpy(temp1,xor(divisor,temp1)); for(i=0;i<s-1;i++)

{

temp1[i] = temp1[i+1];

}

temp1[s-1] = divident[divslen];

}

else

{ for(int i=0;i<divslen;i++) temp2[i] = '0'; strcpy(temp1,xor(temp2, temp1)); for(i=0;i<s-1;i++)

{

temp1[i] = temp1[i+1];

}

temp1[s-1] = divident[divslen];

}

divslen += 1;

} if (temp1[0] == '1')

{

strcpy(temp1,xor(divisor, temp1)); for(i=0;i<s-1;i++)

{

temp1[i] = temp1[i+1];

}

temp1[s-1] = divident[divslen];

}

else

{ for(int i=0;i<divslen;i++) temp2[i] = '0'; strcpy(temp1,xor(temp2,temp1)); for(i=0;i<s-1;i++)

{

temp1[i] = temp1[i+1];

}

temp1[s-1] = divident[divslen];

}

return temp1;

} void printList(int listno) { if(listno==1)

{

n \*temp = head1;

printf("\n");

while(temp != NULL)

{

printf("x^%d + ",temp->power); temp = temp->next;

}

} if(listno==2)

{

n \*temp = head2;

printf("\n");

while(temp != NULL)

{

printf("x^%d + ",temp->power); temp = temp->next;

}

} if(listno==3)

{

n \*temp = head;

printf("\n");

while(temp != NULL)

{

printf("x^%d + ",temp->power); temp = temp->next;

}

} } void polydiv(int t1 , int t2)

{ int i=0,j=0,t=t1+t2; printf("\n\nMessage :"); for(i=0;i<t1;i++)

{

n \*Poly1 = (n\*) malloc(sizeof(n)); printf("\nEnter the power of term %d :",i+1); scanf("%d",&Poly1->power); n \*temp = head1; if(head1 == NULL)

{

Poly1->next = head1; head1 = Poly1;

}

else

{

while(temp->next != NULL) temp=temp->next;//while loop goes to last element of list temp->next=Poly1;//last element now points to new node

Poly1->next = NULL;

}

} printf("\n\nFor Key :"); for(i=0;i<t2;i++)

{

n \*Poly2 = (n\*) malloc(sizeof(n)); printf("\nEnter the power of term %d :",i+1); scanf("%d",&Poly2->power); n \*temp = head2; if(head2 == NULL)

{

Poly2->next = head2; head2 = Poly2;

}

else

{

while(temp->next != NULL) temp=temp->next;//while loop goes to last element of list temp->next=Poly2;//last element now points to new node

Poly2->next = NULL;

}

}

printf("\n\nMessage :"); printList(1); printf("\n\nKey :"); printList(2); n \*temp1 = head1; while(temp1 != NULL)

{

n \*temp2 = head2; n \*temp3 = head2->next; while(temp2 != temp3)

{

n \*Poly = (n\*) malloc(sizeof(n));

Poly->power = temp1->power + temp2->power; if(head == NULL)

{

Poly->next = head; head = Poly;

}

else

{

n \*temp = head; while(temp->next != NULL)

temp=temp->next;//while loop goes to last element of list temp->next=Poly;//last element now points to new node

Poly->next = NULL;

}

temp2 = temp2->next;

}

temp1 = temp1->next;

}

printf("\n\nTemporary Message : "); printList(3); char data[100],key[100]; n \*dat = head; int m,p,l;

m = head->power; l = head->power; i=0; while(i<=l)

{

p = dat->power; if(p == m)

{ data[i]='1'; if(dat->next != NULL) dat = dat->next;

}

else

data[i]='0';

m--; i++;

} data[i] = '\0'; dat = head2; m = head2->power; l = head2->power; i=0; while(i<=l)

{

p = dat->power; if(p == m)

{ key[i]='1'; if(dat->next != NULL) dat = dat->next;

}

else

key[i]='0'; m--; i++;

} key[i] = '\0'; int keylen,datalen; keylen = strlen(key); datalen = strlen(data); l = keylen+datalen-1; char rem[keylen-1]; char code[datalen]; strcpy(rem,divi(data,key)); printf("\n\t\t\t\tSender's Side : "); printf("\n\nData : %s",data); printf("\nKey : %s",key); printf("\nRemainder : %s",rem); strcpy(code,data); for(i=datalen-keylen+1,j=0;j<keylen-1;i++,j++) code[i] = rem[j]; code[i] = '\0'; strcpy(rem,divi(code,key)); printf("\n\n\t\t\t\tReceivers's Side : "); printf("\n\nEncoded Data : %s",code); printf("\nKey : %s",key); printf("\nRemainder : %s",rem);

}

int main()

{ int t1 = 0 , t2 = 0; printf("Enter the number of terms of Divident : "); scanf("%d",&t1); printf("Enter the number of terms of Divisor : "); scanf("%d",&t2); polydiv(t1,t2); return 0;

}

OUTPUT:

Enter the number of terms of Divident : 4 Enter the number of terms of Divisor : 2 Sender’s Data :

Enter the power of term 1 :6

Enter the power of term 2 :4

Enter the power of term 3 :3

Enter the power of term 4 :0 For Key:

Enter the power of term 1 :3

Enter the power of term 2 :0 Sender’s Data:

x^6 + x^4 + x^3 + x^0 + Key: x^3 + x^0 + Temporary Message: x^9 + x^7 + x^6 + x^3 +

Sender's Side :

Data: 1011001000

Key: 1001

Remainder: 011

Receivers' Side :

Encoded Data: 1011001011

Key: 1001

Remainder: 000

No ERRORS found!!