/\*linked list copy,rev(other\_way)\*/

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node\*link;

};

struct node\*header;

struct node\*header1;

struct node\*create\_sll(struct node\*);

struct node\*display(struct node\*);

struct node\*copy(struct node\*,struct node\*);

struct node\*reversal(struct node\*);

int main()

{

int ch;

while(ch!=5)

{

printf("MAIN MENU\n");

printf("1.create a list\n2.display\n3.copy the list in another list\n4.reverse the list\n5.exit\n");

printf("enter your choice\n");

scanf("%d",&ch);

switch(ch)

{

case 1:header=create\_sll(header);

break;

case 2:header=display(header);

break;

case 3:header1=copy(header,header1);

break;

case 4:header=reversal(header);

break;

case 5:exit(0);

default:

printf("invalid choice\n");

}

}

}

struct node\*create\_sll(struct node\*header)

{

struct node\*new\_node,\*ptr;

int item;

printf("enter -1 to end\n");

printf("enter the data: \n");

scanf("%d",&item);

while(item!=-1)

{

new\_node=(struct node\*)malloc(sizeof(struct node));

new\_node->data=item;

if(header==NULL)

{

new\_node->link=NULL;

header=new\_node;

}

else

{

ptr=header;

while(ptr->link!=NULL)

{

ptr=ptr->link;

}

ptr->link=new\_node;

new\_node->link=NULL;

}

printf("enter the data: \n");

scanf("%d",&item);

}

return header;

}

struct node\*display(struct node\*header)

{

printf("the list is below\n");

struct node\*ptr;

ptr=header;

while(ptr!=NULL)

{

printf("%d\n",ptr->data);

ptr=ptr->link;

}

return header;

}

struct node\*copy(struct node\*header,struct node\*header1)

{

struct node\*new\_node,\*ptr,\*ptr1;

new\_node=(struct node\*)malloc(sizeof(struct node));

ptr=header;

if(header1==NULL)

{

new\_node->data=ptr->data;

new\_node->link=NULL;

header1=new\_node;

ptr=ptr->link;

}

else

{

ptr1=header1;

while(ptr!=NULL)

{

new\_node->data=ptr->data;

ptr->link=new\_node;

ptr1=new\_node;

ptr=ptr->link;

}

}

printf("list copied\n");

return header1;

}

struct node\*reversal(struct node\*header)

{

struct node\*r,\*s; //here,header=q

r=NULL;

s=NULL;

if(header!=NULL)

{

r=header;

s=header->link;

header=header->link;

r->link=NULL;

} //to make 1st node the last node

while(header!=NULL)

{

header=header->link;

s->link=r;

r=s;

s=header;

}

header=r; //to lonking out the last node

printf("list reversed\n");

return header;

}

