/\*

Title : Data\_Structure\_Assignment1

Name : Detender Singh

Date : 9th Oct,2016

Description : Program in C language to store Data in linked lists.

it takes data from the keyboard in the form "example,6"and saves this into the

linked list on the position which is mentioned after ',',for instance, on 6th

node as given here,if data entered without ',' then that data will be stored in

the last node and this process keeps on until user enter'$' after this each node

will be printed on the screen.

\*/

//header files

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

/\*structure defined for linked list

typedef is used here- to use SaveData to initialise the variable, instead of typing long line - struct typeSaveData

\*/

typedef struct typeSaveData{

//variable to save array of chararcters only upto 30.

char info[31];

//pointer to point to the next Node of same struct type

struct typeSaveData\* pointNext;

}SaveData;

//declaring global variables

SaveData \*head = NULL;

//function prototypes

SaveData\* createList(char\* data);

void appendIntoList(char\* data, int loc,int positionTemp);

void printList(void);

/\*

main function : from here the program execution will begin

parameters : nothing

return : int type

\*/

int main(void)

{

/\*initialising the local variables\*/

//variable to take keyboard input

char dataVar[31] = "";

//to store data

char\* name;

//to store position

char\* listNo;

//variables used to set conditions

char dollar = '$';

int list = 0;

int flag = 1;

int loopContinue = 1;

int position = 1;

//it will keep on taking input and saving that data into list until it detects '$' character

while(loopContinue==1)

{

printf("enter string with position after comma,to save at last position do not enter ',', to end list enter '$'\n");

//taking input from the keyboard using fgets function

fgets(dataVar, sizeof(dataVar), stdin);

//parsing the data using strtok and storing data and position in different variables

name = strtok(dataVar,",\n");

listNo = strtok(NULL,"\n");

//incrementing position vaiable to keep the track of total no. of nodes

position++;

//declaring and assigning length of the value saved in name string to the len variable

size\_t len = strlen(name);

//checking for '$' character by traversing through name string

int i;

for(i = 0; i < len; i++)

{

if(name[i] == dollar)

{

//if '$' is present in the string then reset loopContinue variable, so that it stops taking further input

loopContinue = 0;

break;

}

}

//this if is just making sure that when it detects '$' character,it must not save data that is having this character

if(loopContinue == 1)

{

/\*this if is checking whether user entered position or not,because if no position has been entered then

listNo string will have NULL value.

\*/

if(listNo!=NULL)

{

//converting ascii digits to integer value,if any character other than digits entered then it retruns 0

list = atoi(listNo);

}

else//if nothing is in listNo string

{

list = -1;

}

/\*this if is making sure that user has entered only digit characters after ','

because if not than atoi function will return 0.\*/

if(list !=0)

{

//this if is checking, is this the first data that user has entered or not

if(flag==1)

{

//if first data entered then call function to create list

createList(name);

//now list created so reset flag

flag =0;

}

else

{

//if user entered position which is not yet available or created then

if(list>=position)

{

printf("\nposition [%d] not available yet\n", list);

//decrementing position because nothing saved during this time

position--;

}

else//else function call to append data into the list

{

appendIntoList(name,list,position);

}

}

}

else//if user entered character other than digits

{

printf("please enter only digit after','\n");

//decrementing position because nothing saved during this time

position--;

}

}

}

//once user entered '$' character now function call to print all nodes

printList();

return 0;

}

/\*

Function : createList()

-- this function creates head node saves the value into the head node.

parameters : char\* data - pointer of character type of the value to be saved.

returns : SaveData\* ptr - pointer of SaveData type,a structure

\*/

SaveData\* createList(char\* data)

{

printf("\n creating list with headnode as [%s]\n",data);

//declaring and allocating memory to the ptr pointer variable

SaveData \*ptr = (SaveData\*)malloc(sizeof(SaveData));

/\*if there is no memory available then malloc function will return NULL

which will be assigned to ptr, so this condition is checking for memory

available to store head node\*/

if(NULL == ptr)

{

printf("\n Node creation failed \n");

return NULL;

}

/\*to save the data into node, strncpy command is used, to be sure it should not

overfill the info character array, the size of this info array is measured and then

assigned to variable destination\_size \*/

size\_t destination\_size = sizeof(ptr->info);

strncpy(ptr->info, data, destination\_size);

//saving '\0' in the last of info character because strncpy command does not put this character automatically

ptr->info[destination\_size - 1] = '\0';

//setting pointer in the node = NULL

ptr->pointNext = NULL;

//saving the address available in ptr into head,making this a head node

head = ptr;

return ptr;

}

/\*

Function :appendIntoList()

-- this function saves the value into the list at given position and adjust rest of

the list positions and head according to the appended value.

parameters : char\* data - pointer of character type of the value to be saved.

int loc - integer type variable which will get either -1 or +integer value.

int position - integer type variable which used to save the total no. of nodes

at the moment when function called.

returns : nothing

\*/

void appendIntoList(char\* data, int loc,int positionTemp)

{

//variable declerations

int i;

int last;

SaveData \*temp,\*left,\*right;

//assigning address which head is having, to the right variable

right=head;

/\*if user did not enter the position it will assign value = positionTemp-1 into the last variable otherwise

it will assign value entered in the loc variable into last variable

\*/

if(loc != -1)

{

last = loc;

}

else

{

last =positionTemp-1;

}

/\*after getting the position into last variable it goes into for loop

to reach upto the position where entered value need to be saved,

so left pointer pointing to the previous position and right pointing to

the position where the value need to be saved

\*/

for(i=1;i<last;i++)

{

left=right;

right=right->pointNext;

}

//allocating memory to temp variable

temp=(SaveData\*)malloc(sizeof(SaveData));

//checking for memory available or not

if(NULL == temp)

{

printf("\n Node creation failed \n");

}

//using strncpy command to save data into temp->info

size\_t destination\_size = sizeof(temp->info);

strncpy(temp->info, data, destination\_size);

//saving '\0' in the last of info character because strncpy command does not put this character automatically

temp->info[destination\_size - 1] = '\0';

//if position where data need to be entered is not the head node

if(last>1)

{

/\*previous position i.e. left, will point to the new temp node and now left assigned with temp

and this new left will point to right.

\*/

left->pointNext=temp;

left=temp;

left->pointNext=right;

}

else//else data need to be stored in the head node

{

/\*it will be saved into the left, pointing to the head node

and then left will become head

\*/

left=temp;

left->pointNext=head;

head =left;

}

return;

}

/\*Function: printList()

- this function prints the whole list one by one

parameters: SaveData\* current -which takes the node from where it will start printing the list upto end

return : Nothing

\*/

void printList(void)

{

// variable is used to print serial no. before each character of the list

int serialNo = 1;

//local variable ptr,assigned with the head node

SaveData\* ptr = head;

printf("your saved list:\n");

//this loop will traverse upto the last node and prints the nodes side by side

while(ptr!=NULL)

{

printf("%d. %s\n",serialNo,ptr->info);

ptr=ptr->pointNext;

serialNo++;

}

}

//output on next page

