22BCE5252

ANIRUDH SRIDHAR

OOPS ASSIGNMENT- 3

1)

Code:

#include <stdio.h>

void leftshift(int \*arr, int n, int len) {

int i;

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

printf("%d ", arr[i]);

}

printf("\n");

for (i= 0; i<n; i++) {

int j;

int first= arr[0];

for (j= 0; j<len-1; j++) {

arr[j]= arr[j+1];

}

arr[j]= first;

for (int k= 0; k<len; k++) {

printf("%d ", arr[k]);

}

printf("\n");

}

}

int main() {

int arr[]= {11, 22, 33, 44, 55};

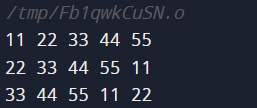
int len= sizeof(arr)/sizeof(arr[0]);

leftshift(arr, 2, len);

return 0;

}

Output:



2)

Code:

#include <stdio.h>

int row\_check(int arr[9][9], int n) {

int c[9]= {0};

for (int i= 0; i < 9; i++) {

c[arr[n][i]]++;

}

int flag= 1;

for (int i= 0; i < 9; i++) {

if ((c[i] > 1) || (c[i] == 0)) {

flag= 0;

}

}

return flag;

}

int column\_check(int arr[9][9], int n) {

int c[9]= {0};

for (int i= 0; i < 9; i++) {

c[arr[i][n]]++;

}

int flag= 1;

for (int i= 0; i < 9; i++) {

if ((c[i] > 1) || (c[i] == 0)) {

flag= 0;

}

}

return flag;

}

void number(int n, int arr[]) {

switch(n) {

case(1):

arr[0]= 0;

arr[1]= 3;

break;

case(2):

arr[0]= 3;

arr[1]= 6;

break;

case(3):

arr[0]= 6;

arr[1]= 9;

break;

}

}

int matrix\_check(int arr[9][9], int start, int end) {

int c[9]= {0};

for (int i= start; i < end; i++) {

for (int j= start; j < end; j++) {

c[arr[i][j]]++;

}

}

int flag= 1;

for (int i= 0; i < 9; i++) {

if ((c[i] > 1) || (c[i] == 0)) {

flag= 0;

}

}

return flag;

}

void sudoku\_check(int arr[9][9]) {

int flag= 1;

for (int i= 0; i < 9; i++) {

if (!row\_check(arr, i)) {

flag= 0;

}

if (!column\_check(arr, i)) {

flag= 0;

}

}

for (int i= 1; i < 4; i++) {

int a[2];

number(i, a);

if (!matrix\_check(arr, a[0], a[1])) {

flag= 0;

}

}

if (flag) {printf("Correct Solution");}

else {printf("Wrong Solution");}

}

int main() {

int arr[9][9] = {{5,1,7,2,6,4,8,9,3},{9,2,6,8,3,5,7,4,1},{4,8,3,9,7,1,5,6,2},{1,3,5,4,9,6,2,8,7},{7,9,2,5,1,8,4,3,6},{8,6,4,3,2,7,9,1,5},{3,7,8,6,4,2,1,5,9},{2,5,9,1,8,3,6,7,4},{6,4,1,7,5,9,3,2,6}};

sudoku\_check(arr);

return 0;

}

Output:



3)

Code:

#import <stdio.h>

void lcs(char a1[], char a2[], int l1, int l2) {

int r;

if (l1 > l2) {

r= l1;

}

else {

r= l2;

}

char res[r];

int k= 0;

for (int i= 0; i<l1; i++) {

for (int j= 0; j<l2; j++) {

if (a1[i] == a2[j]) {

res[k++] = a1[i];

}

}

}

printf("%s", res);

}

int main() {

int l1, l2;

scanf("%d %d", &l1, &l2);

char str1[l1];

printf("Enter string 1\n");

scanf("%s", &str1);

char str2[l2];

printf("Enter string 2\n");

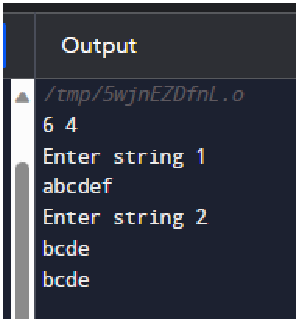
scanf("%s", &str2);

lcs(str1, str2, l1, l2);

return 0;

}

Output:



4)

Code:

#include <stdio.h>

#include <string.h>

void fibonacci(int n) {

char x[255]= "a", z[255];

char y[255]= "b";

if (n == 0) {

printf("a");

}

else if (n == 1) {

printf("b");

}

else {

for (int i= 0; i<n; i++) {

printf("%s\n", x);

strcpy(z, x);

strcpy(x, y);

strcat(y, z);

}

}

}

int main() {

int n;

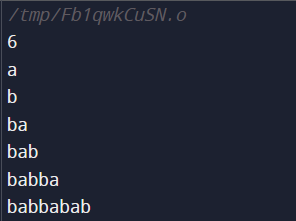
scanf("%d", &n);

fibonacci(n);

return 0;

}

Output:



5)

Code:

#include <stdio.h>

void toH(int n, char rodA, char rodC, char rodB)

{

if (n == 1)

{

printf("\n disk 1 move from: rod %c to rod %c",rodA ,rodC );

return;

}

toH(n-1, rodA, rodB, rodC);

printf("\n disk %d moves from: rod %c to rod %c", n, rodA, rodC);

toH(n-1, rodB, rodC,rodA);

}

int main()

{

int n;

printf("Enter number of disks: ");

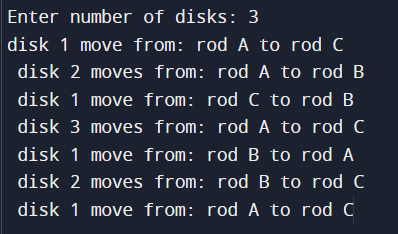
scanf("%d", &n);

toH(n, 'A','C','B');

return 0;

}

Output:



6)

Code:

#include <stdio.h>

void rev();

int main() {

printf("Enter a sentence: ");

rev();

return 0;

}

void rev() {

char c;

scanf("%c", &c);

if (c != '\n') {

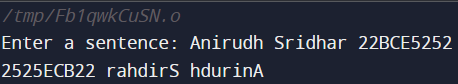
rev();

printf("%c", c);

}

}

Output:



7)

Code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int width = 80, year;

int cols, lead, gap;

const char \*wd[] = { "Su", "Mo", "Tu", "We", "Th", "Fr", "Sa" };

struct months {

const char \*name;

int days, start\_wday, at;

} months[12] = {

{ "January", 31, 0, 0 },

{ "February", 28, 0, 0 },

{ "March", 31, 0, 0 },

{ "April", 30, 0, 0 },

{ "May", 31, 0, 0 },

{ "June", 30, 0, 0 },

{ "July", 31, 0, 0 },

{ "August", 31, 0, 0 },

{ "September", 30, 0, 0 },

{ "October", 31, 0, 0 },

{ "November", 30, 0, 0 },

{ "December", 31, 0, 0 }

};

void space(int n) { while (n-- > 0) putchar(' '); }

void init\_months()

{

int i;

if ((!(year % 4) && (year % 100)) || !(year % 400))

months[1].days = 29;

year--;

months[0].start\_wday

= (year \* 365 + year/4 - year/100 + year/400 + 1) % 7;

for (i = 1; i < 12; i++)

months[i].start\_wday =

(months[i-1].start\_wday + months[i-1].days) % 7;

cols = (width + 2) / 22;

while (12 % cols) cols--;

gap = cols - 1 ? (width - 20 \* cols) / (cols - 1) : 0;

if (gap > 4) gap = 4;

lead = (width - (20 + gap) \* cols + gap + 1) / 2;

year++;

}

void print\_row(int row)

{

int c, i, from = row \* cols, to = from + cols;

space(lead);

for (c = from; c < to; c++) {

i = strlen(months[c].name);

space((20 - i)/2);

printf("%s", months[c].name);

space(20 - i - (20 - i)/2 + ((c == to - 1) ? 0 : gap));

}

putchar('\n');

space(lead);

for (c = from; c < to; c++) {

for (i = 0; i < 7; i++)

printf("%s%s", wd[i], i == 6 ? "" : " ");

if (c < to - 1) space(gap);

else putchar('\n');

}

while (1) {

for (c = from; c < to; c++)

if (months[c].at < months[c].days) break;

if (c == to) break;

space(lead);

for (c = from; c < to; c++) {

for (i = 0; i < months[c].start\_wday; i++) space(3);

while(i++ < 7 && months[c].at < months[c].days) {

printf("%2d", ++months[c].at);

if (i < 7 || c < to - 1) putchar(' ');

}

while (i++ <= 7 && c < to - 1) space(3);

if (c < to - 1) space(gap - 1);

months[c].start\_wday = 0;

}

putchar('\n');

}

putchar('\n');

}

void print\_year()

{

int row;

char buf[32];

sprintf(buf, "%d", year);

space((width - strlen(buf)) / 2);

printf("%s\n\n", buf);

for (row = 0; row \* cols < 12; row++)

print\_row(row);

}

int main(int c, char \*\*v)

{

int i, year\_set = 0;

printf("Input a valid year: ");

scanf("%d", &year);

for (i = 1; i < c; i++) {

if (!strcmp(v[i], "-w")) {

if (++i == c || (width = atoi(v[i])) < 20)

printf("\n Incorrect month");

} else if (!year\_set) {

if (!sscanf(v[i], "%d", &year) || year <= 0)

year = 1969;

year\_set = 1;

} else

printf("\n Incorrect data");

}

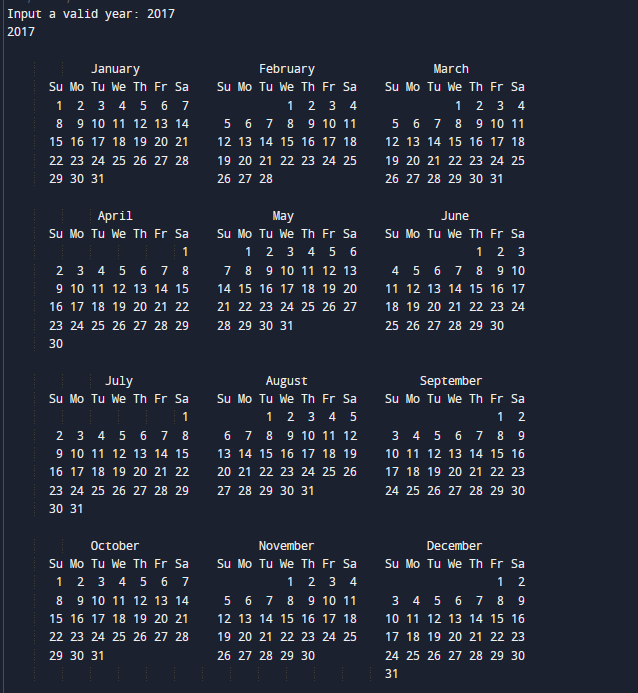
init\_months();

print\_year();

return 0;

}

Output:



8)

Code:

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

char square[10]={'0','1','2','3','4','5','6','7','8','9'};

int choice,player; //Global Variables

int checkforWin(); //Checks if either player has won

void displayBoard(); //Displays the board

void markBoard(char mark); //Marks the X or O on the board

int main(){

int gamestatus;

char mark;

player=1;

do{

displayBoard();

player=(player%2)?1:2;

printf("Player %d, Enter a Number: ",player);

scanf("%d",&choice);

if(player==1){

mark='X';

}

else{

mark='O';

}

markBoard(mark);

gamestatus=checkforWin();

player++;

if(gamestatus==1){

printf("Congrats!. Player 1 has Won!");

}

else if(gamestatus==2){

printf("Congrats!. Player 2 has Won!");

}

else{

printf("It is a Draw!");

}

}while(gamestatus==-1);

return 0;

}

/\*\*

FUNCTION TO RETURN GAME STATUS

1 FOR GAME IS OVER WITH RESULT

0 FOR GAME IS OVER WITHOUT RESULT

-1 FOR GAME STILL IN PROGRESS

\*\*/

int checkforWin(){

int returnVal=0;

if(square[1]=='X' && square[2]=='X' && square[3]=='X'){

returnVal=1;

}

else if(square[4]=='X' && square[5]=='X' && square[6]=='X'){

returnVal=1;

}

else if(square[7]=='X' && square[8]=='X' && square[9]=='X'){

returnVal=1;

}

else if(square[1]=='X' && square[4]=='X' && square[7]=='X'){

returnVal=1;

}

else if(square[2]=='X' && square[5]=='X' && square[8]=='X'){

returnVal=1;

}

else if(square[3]=='X' && square[6]=='X' && square[9]=='X'){

returnVal=1;

}

else if(square[1]=='X' && square[5]=='X' && square[9]=='X'){

returnVal=1;

}

else if(square[3]=='X' && square[5]=='X' && square[7]=='X'){

returnVal=1;

}

else if(square[1]=='O' && square[2]=='O' && square[3]=='O'){

returnVal=2;

}

else if(square[4]=='O' && square[5]=='O' && square[6]=='O'){

returnVal=2;

}

else if(square[7]=='O' && square[8]=='O' && square[9]=='O'){

returnVal=2;

}

else if(square[1]=='O' && square[4]=='O' && square[7]=='O'){

returnVal=2;

}

else if(square[2]=='O' && square[5]=='O' && square[8]=='O'){

returnVal=2;

}

else if(square[3]=='O' && square[6]=='O' && square[9]=='O'){

returnVal=2;

}

else if(square[1]=='O' && square[5]=='O' && square[9]=='O'){

returnVal=2;

}

else if(square[3]=='O' && square[5]=='O' && square[7]=='O'){

returnVal=2;

}

else if(square[1]!='1' && square[2]!='2' && square[3]!='3' && square[4]!='4' && square[5]!='5' && square[6]!='6' && square[7]!='7' && square[8]!='8' && square[9]!='9'){

returnVal=0;

}

else{

returnVal=-1;

}

return returnVal;

}

/\*\*

DISPLAYS THE BOARD

\*\*/

void displayBoard(){

system("cls");

printf("\n\n\tTic Tac Toe\n\n");

printf("Player 1 (X)- Player 2 (O)\n\n\n");

printf("\t\t|\t\t|\n");

printf("\t%c\t|\t%c\t|\t%c\n",square[1],square[2],square[3]);

printf("\t\t|\t\t|\n");

printf("----------------------------------------------\n");

printf("\t\t|\t\t|\n");

printf("\t%c\t|\t%c\t|\t%c\n",square[4],square[5],square[6]);

printf("\t\t|\t\t|\n");

printf("----------------------------------------------\n");

printf("\t\t|\t\t|\n");

printf("\t%c\t|\t%c\t|\t%c\n",square[7],square[8],square[9]);

printf("\t\t|\t\t|\n");

}

/\*\*

Sets the board with the correct character,

X or O in the correct spot in the Array

\*\*/

void markBoard(char mark){

if(choice==1 && square[1]=='1'){

square[1]=mark;

}

else if(choice==2 && square[2]=='2'){

square[2]=mark;

}

else if(choice==3 && square[3]=='3'){

square[3]=mark;

}

else if(choice==4 && square[4]=='4'){

square[4]=mark;

}

else if(choice==5 && square[5]=='5'){

square[5]=mark;

}

else if(choice==6 && square[6]=='6'){

square[6]=mark;

}

else if(choice==7 && square[7]=='7'){

square[7]=mark;

}

else if(choice==8 && square[8]=='8'){

square[8]=mark;

}

else if(choice==9 && square[9]=='9'){

square[9]=mark;

}

else{

printf("Invalid Move");

player--;

getch(); //Makes sure that board doesn't need to be redrawn after an invalid move and can cancel the number typed

}

}

Output:

[?2004l

sh: 1: cls: not found

Tic Tac Toe

Player 1 (X)- Player 2 (O)

| |

1 | 2 | 3

| |

----------------------------------------------

| |

4 | 5 | 6

| |

----------------------------------------------

| |

7 | 8 | 9

| |

Player 1, Enter a Number: 3

sh: 1: cls: not found

It is a Draw!

Tic Tac Toe

Player 1 (X)- Player 2 (O)

| |

1 | 2 | X

| |

----------------------------------------------

| |

4 | 5 | 6

| |

----------------------------------------------

| |

7 | 8 | 9

| |

Player 2, Enter a Number: 2

sh: 1: cls: not found

It is a Draw!

Tic Tac Toe

Player 1 (X)- Player 2 (O)

| |

1 | O | X

| |

----------------------------------------------

| |

4 | 5 | 6

| |

----------------------------------------------

| |

7 | 8 | 9

| |

Player 1, Enter a Number: 5

sh: 1: cls: not found

It is a Draw!

Tic Tac Toe

Player 1 (X)- Player 2 (O)

| |

1 | O | X

| |

----------------------------------------------

| |

4 | X | 6

| |

----------------------------------------------

| |

7 | 8 | 9

| |

Player 2, Enter a Number: 7

sh: 1: cls: not found

It is a Draw!

Tic Tac Toe

Player 1 (X)- Player 2 (O)

| |

1 | O | X

| |

----------------------------------------------

| |

4 | X | 6

| |

----------------------------------------------

| |

O | 8 | 9

| |

Player 1, Enter a Number: 6

sh: 1: cls: not found

It is a Draw!

Tic Tac Toe

Player 1 (X)- Player 2 (O)

| |

1 | O | X

| |

----------------------------------------------

| |

4 | X | X

| |

----------------------------------------------

| |

O | 8 | 9

| |

Player 2, Enter a Number: 4

sh: 1: cls: not found

It is a Draw!

Tic Tac Toe

Player 1 (X)- Player 2 (O)

| |

1 | O | X

| |

----------------------------------------------

| |

O | X | X

| |

----------------------------------------------

| |

O | 8 | 9

| |

Player 1, Enter a Number: 9

Congrats!. Player 1 has Won!

9)

Code:

#include <stdio.h>

#include <string.h>

// Get the number of strings and its corresponding input string from the user. Sort the strings

// using bubble sort.

/\*

Algorithm:

- Get strings in a 2D string array

- Somehow, sort the strings using bubble sort.. is it possible to assign an integer value to each string, use that

to sort the strings? Alternative: use strcmp, check if (strcmp(char1,char2)>0) or <0 for comparison.

\*/

void bubbleSort(char arr[], int size) {

for (int step = 0; step < size - 1; ++step) {

for (int i = 0; i < size - step - 1; ++i) {

if ((strcmp(arr[i], arr[i+1]))>0) {

char\* temp = arr[i];

arr[i] = arr[i + 1];

arr[i + 1] = temp;

}

}

}

}

// print array

void printarr(char arr[], int size) {

for (int i = 0; i < size; ++i) {

printf("%s ", arr[i]);

}

printf("\n");

}

int main() {

printf("enter limit:");

int n;

scanf("%d", &n);

char strings[n][20];

for (int i=0;i<n;i++){

printf("\n enter string (max size 20): ");

scanf("%s",&strings[i]);

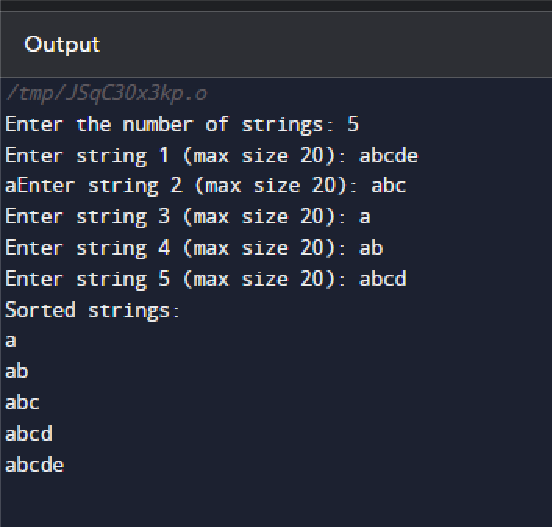
}

bubbleSort(strings,n);

printarr(strings,n);

}

Output:



10)

Code:

#include <stdio.h>

#include <string.h>

int main() {

char txt[]= "Test text line";

int len= strlen(txt);

char word[20];

int words= 0;

int i, j= 0;

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

if (txt[i] == ' ') {

words++;

word[j] = '\0';

printf("%s\n", word);

j= 0;

}

else {

word[j] = txt[i];

j++;

}

}

word[j] = '\0';

printf("%s\n", word);

return 0;

}

Output:

