

DATA STRUCTURES LAB FILE

Submitted by VINAYAK GOSWAMI (2018UCO1598)



2018UCO1598

COE SECTION-2

# **Matrix Multiplication**

CODE:

//BEGIN

#include <iostream>

using namespace std;

#define N 3

void multiply(int mat1[][N],int mat2[][N],int res[][N])

{

int i, j, k;

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

for (j = 0; j < N; j++){

res[i][j] = 0;

for (k = 0; k < N; k++)

res[i][j] += mat1[i][k] \* mat2[k][j];} } }

int main(){

int i, j;

int res[N][N], mat1[N][N], mat2[N][N];

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

for(j = 0; j < N; j++)

cin >> mat1[i][j];

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

for(j = 0; j < N; j++)

cin >> mat2[i][j];

multiply(mat1, mat2, res);

cout << "Result matrix is \n";

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

for (j = 0; j < N; j++)

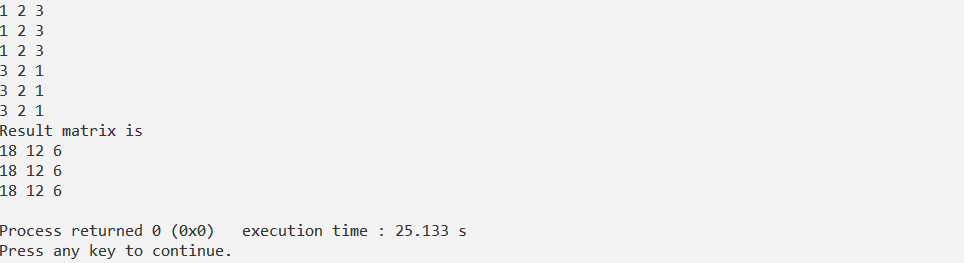
cout << res[i][j] << " ";

cout << "\n";}

return 0;}

//END

OUTPUT:



# **To Print Numbers in English Alphabets**

CODE:

//BEGIN

#include <stdio.h>

#include <string.h>

#include <string>

#include <stdlib.h>

void convert\_to\_words(char \*num){

int len = strlen(num);

if (len == 0) {

fprintf(stderr, "empty string\n");

return;}

if (len > 4) {

fprintf(stderr, "Length more than 4 is not supported\n");

return;}

char \*single\_digits[] = { "zero", "one", "two","three", "four","five","six", "seven", "eight", "nine"};

char \*two\_digits[] = {"", "ten", "eleven", "twelve","thirteen", "fourteen","fifteen", "sixteen","seventeen", "eighteen", "nineteen"};

char \*tens\_multiple[] = {"", "", "twenty", "thirty", "forty", "fifty","sixty", "seventy", "eighty", "ninety"};

char \*tens\_power[] = {"hundred", "thousand"};

printf("\n%s: ", num);

if (len == 1) {

printf("%s\n", single\_digits[\*num - '0']);

return; }

while (\*num != '\0') {

if (len >= 3) {

if (\*num -'0' != 0) {

printf("%s ", single\_digits[\*num - '0']);

printf("%s ", tens\_power[len-3]); }

--len; }

else {

if (\*num == '1') {

int sum = \*num - '0' + \*(num + 1)- '0';

printf("%s\n", two\_digits[sum]);

return;}

else if (\*num == '2' && \*(num + 1) == '0') {

printf("twenty\n");

return; }

else {

int i = \*num - '0';

printf("%s ", i? tens\_multiple[i]: "");

++num;

if (\*num != '0')

printf("%s ", single\_digits[\*num - '0']); } }

++num; } }

int main(void){

char s[100];

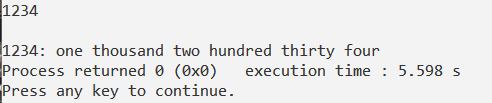
gets(s);

convert\_to\_words(s);

return 0; }

//END

OUTPUT:



# **Fibonacci Numbers using recursion**

CODE:

//BEGIN

#include <iostream>

using namespace std;

int fib(int x){

if((x==1)||(x==0)){

return(x);}

else{

return(fib(x-1)+fib(x-2)); } }

int main(){

int x , i=0;

cout << "Enter the number of terms of series : ";

cin >> x;

cout << "\nFibonnaci Series : ";

while(i < x) {

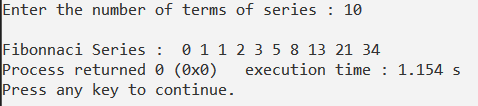
cout << " " << fib(i);

i++; }

return 0; }

//END

OUTPUT:



# **Factorial of a number using recursion**

CODE:

//BEGIN

#include<iostream>

using namespace std;

int factorial(int n) {

if(n > 1)

return n \* factorial(n - 1);

else

return 1; }

int main() {

int n;

cout << "Enter a positive integer: ";

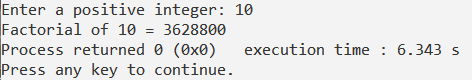
cin >> n;

cout << "Factorial of " << n << " = " << factorial(n);

return 0; }

//END

OUTPUT:



# **File handling: to store data of employees**

CODE:

//BEGIN

#include <iostream>

#include <fstream>

#define FILE\_NAME "emp.dat"

using namespace std;

class Employee {

private :

int empID; char empName[100] ;char designation[100];int ddj,mmj,yyj;int ddb,mmb,yyb;

public :

void readEmployee() {

cout<<"EMPLOYEE DETAILS"<<endl;

cout<<"ENTER EMPLOYEE ID : " ;

cin>>empID;

cin.ignore(1);

cout<<"ENTER NAME OF THE EMPLOYEE : ";

cin.getline(empName,100);

cout<<"ENTER DESIGNATION : ";

cin.getline(designation,100);

cout<<"ENTER DATE OF JOIN:"<<endl;

cout<<"DATE : "; cin>>ddj;

cout<<"MONTH: "; cin>>mmj;

cout<<"YEAR : "; cin>>yyj;

cout<<"ENTER DATE OF BIRTH:"<<endl;

cout<<"DATE : "; cin>>ddb;

cout<<"MONTH: "; cin>>mmb;

cout<<"YEAR : "; cin>>yyb;}

void displayEmployee() {

cout<<"EMPLOYEE ID: "<<empID<<endl

<<"EMPLOYEE NAME: "<<empName<<endl

<<"DESIGNATION: "<<designation<<endl

<<"DATE OF JOIN: "<<ddj<<"/"<<mmj<<"/"<<yyj<<endl

<<"DATE OF BIRTH: "<<ddb<<"/"<<mmb<<"/"<<yyb<<endl; }

};

int main(){

Employee emp;

emp.readEmployee();

fstream file;

file.open(FILE\_NAME,ios::out|ios::binary);

if(!file){

cout<<"Error in creating file...\n";

return -1; }

file.write((char\*)&emp,sizeof(emp));

file.close();

cout<<"Date saved into file the file.\n";

file.open(FILE\_NAME,ios::in|ios::binary);

if(!file){

cout<<"Error in opening file...\n";

return -1; }

if(file.read((char\*)&emp,sizeof(emp))){

cout<<endl<<endl;

cout<<"Data extracted from file..\n";

emp.displayEmployee(); }

else{

cout<<"Error in reading data from file...\n";

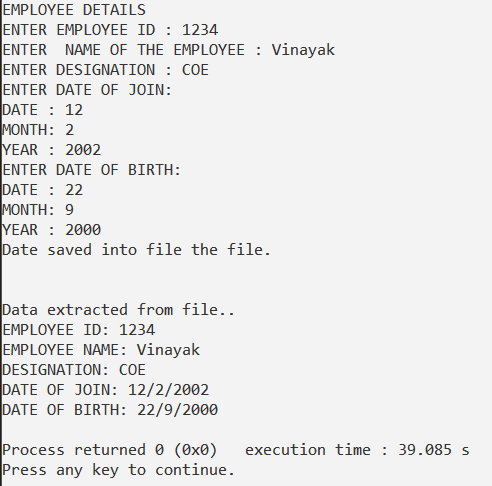
return -1; }

file.close();

return 0; }

//END

OUTPUT:



# **STACKS**

CODE:

//BEGIN

#include<bits/stdc++.h>

using namespace std;

#define MAX 1000

class Stack {

int top;

public:

int a[MAX];

Stack() { top = -1; }

bool push(int x);

int pop();

bool isEmpty();

void show(); };

void Stack::show() {

if(isEmpty())

cout<<"\nNothing to show";

else

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

cout<<a[i]<<"\n"; }

bool Stack::push(int x) {

if (top >= (MAX-1)) {

cout << "Stack Overflow";

return false; }

else {

a[++top] = x;

cout<<x <<" pushed into stack\n";

return true; } }

int Stack::pop() {

if (top < 0) {

cout << "Stack Underflow";

return 0; }

else {

int x = a[top--];

return x; } }

bool Stack::isEmpty() {

return (top < 0); }

int main() {

class Stack s; int n;

do {

cin>>n;

switch(n) {

case 1: {int a;cin>>a;s.push(a); break;}

case 2: {s.pop(); break;}

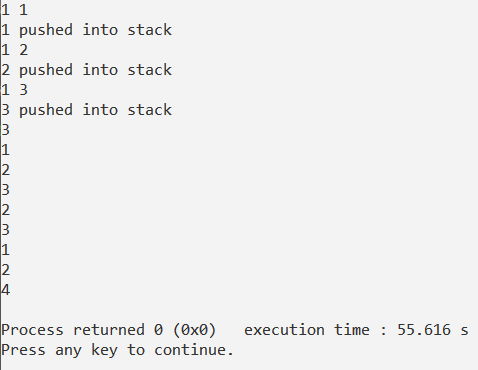
case 3: {s.show(); break;}

default:{break;} } }while(n!=4);

return 0; }

//END

OUTPUT:



# **QUEUES**

CODE:

//BEGIN

#include <iostream>

using namespace std;

int queue[100], n = 100, front = - 1, rear = - 1;

void Insert() {

int val;

if (rear == n - 1)

cout<<"Queue Overflow"<<endl;

else {

if (front == - 1)

front = 0;

cout<<"Insert the element in queue : "<<endl;

cin>>val;

rear++;

queue[rear] = val; } }

void Delete() {

if (front == - 1 || front > rear) {

cout<<"Queue Underflow ";

return ; }

else {

cout<<"Element deleted from queue is : "<< queue[front] <<endl;

front++;; } }

void Display() {

if (front == - 1)

cout<<"Queue is empty"<<endl;

else {

cout<<"Queue elements are : ";

for (int i = front; i <= rear; i++)

cout<<queue[i]<<" ";

cout<<endl; } }

int main() {

int ch;

cout<<"1) Insert element to queue"<<endl; cout<<"2) Delete element from queue"<<endl;

cout<<"3) Display all the elements of queue"<<endl; cout<<"4) Exit"<<endl;

do {

cout<<"Enter your choice : "<<endl; cin>>ch;

switch (ch) {

case 1: Insert(); break;

case 2: Delete(); break;

case 3: Display(); break;

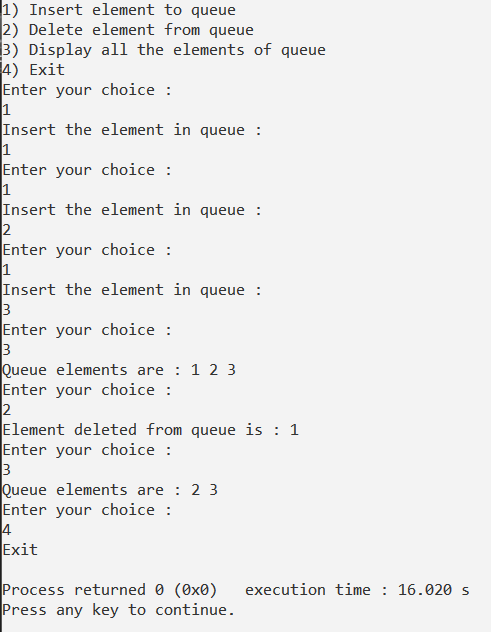
case 4: cout<<"Exit"<<endl; break;

default: cout<<"Invalid choice"<<endl; } }while(ch<4 && ch>0);

return 0; }

//END

OUTPUT:



# **LINKED LIST**

CODE:

//BEGIN

#include <iostream>

using namespace std;

struct Node {

int data;

struct Node \*next; };

struct Node\* head = NULL;

void insert(int new\_data) {

struct Node\* new\_node = new Node;

new\_node->data = new\_data;

new\_node->next = head;

head = new\_node; }

void display() {

struct Node\* ptr;

ptr = head;

while (ptr != NULL) {

cout<< ptr->data <<" "; ptr = ptr->next; } }

int main() {

int ch;

cout<<"1) Insert element to queue"<<endl; cout<<"2) Display all the elements of queue"<<endl;

cout<<"3) Exit"<<endl;

do {

cout<<"Enter your choice : "<<endl; cin>>ch;

switch (ch) {

case 1:{int n;cin>>n;insert(n); break;}

case 2: display(); break;

case 3: cout<<"Exit"<<endl; break;

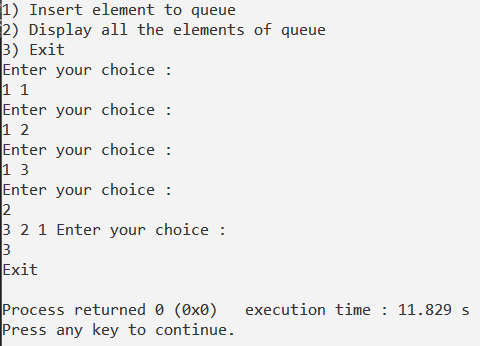
default: cout<<"Invalid choice"<<endl; }

}while(ch<3 && ch>0);

return 0; }

//END

OUTPUT:



# **CLASS TO HOLD DATA OF EMPLOYES**

CODE:

//BEGIN

#include <iostream>

using namespace std;

class Employee {

private :

int empID; char empName[100] ; char designation[100]; int ddj,mmj,yyj;

int ddb,mmb,yyb;

public :

void readEmployee() {

cout<<"EMPLOYEE DETAILS"<<endl; cout<<"ENTER EMPLOYEE ID : " ;

cin>>empID; cin.ignore(1);

cout<<"ENTER NAME OF THE EMPLOYEE : "; cin.getline(empName,100);

cout<<"ENTER DESIGNATION : "; cin.getline(designation,100);

cout<<"ENTER DATE OF JOIN:"<<endl; cout<<"DATE : "; cin>>ddj;

cout<<"MONTH: "; cin>>mmj; cout<<"YEAR : "; cin>>yyj;

cout<<"ENTER DATE OF BIRTH:"<<endl; cout<<"DATE : "; cin>>ddb;

cout<<"MONTH: "; cin>>mmb; cout<<"YEAR : "; cin>>yyb; }

void displayEmployee() {

cout<<"EMPLOYEE ID: "<<empID<<endl

<<"EMPLOYEE NAME: "<<empName<<endl

<<"DESIGNATION: "<<designation<<endl

<<"DATE OF JOIN: "<<ddj<<"/"<<mmj<<"/"<<yyj<<endl

<<"DATE OF BIRTH: "<<ddb<<"/"<<mmb<<"/"<<yyb<<endl; } };

int main() {

Employee emp;

emp.readEmployee();

emp.displayEmployee();

return 0; }

//END

OUTPUT:

