DSA LAB 5

**TASK1:**

**CODE:**

#include<iostream>

#include<string>

using namespace std;

class Stack {

int size=10;

char arr[10];

int top;

public:

//constructor

Stack():top(0){}

//is empty function

bool isEmpty()

{

return top == 0;

}

//is full fuunction

bool isFull()

{

return top == 9;

}

//push value function

void push(char x)

{

if (isFull())

{

cout << "Stack is full! bye bye";

return;

}

arr[top] = x;

top++;

}

//peek value function

void peek(char &x)

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

x = arr[top];

}

//peek value function

void pop()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

top--;

}

void show()

{

for (int i = 0; i < 10; i++)

{

cout << arr[i] << " ";

}

}

};

int main()

{

Stack stack;

char ch;

while (1)

{

system("cls");

cout << "1. insert value\n2. pop value\n3. peek value\n4. is Empty\n5. is Full\n6. show stack \n7. exit\n";

int num;

cin >> num;

switch (num)

{

case 1:

cout << "\nEnter character: ";

cin >> ch;

stack.push(ch);

system("pause");

break;

case 2:

stack.pop();

system("pause");

break;

case 3:

stack.peek(ch);

cout << "character peek: " << ch;

system("pause");

break;

case 4:

stack.isEmpty();

system("pause");

break;

case 5:

stack.isFull();

system("pause");

break;

case 6:

stack.show();

system("pause");

break;

case 7:

exit(0);

system("pause");

break;

default:

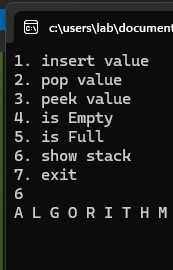
break;

}

}

}

**OUTPUT:**



**TASK2:**

**CODE:**

**Part A**

#include<iostream>

#include<string>

using namespace std;

class Stack {

int size=10;

int arr[10] = { 0 };

int top;

public:

//constructor

Stack():top(0){}

//is empty function

bool isEmpty()

{

return top == 0;

}

//is full fuunction

bool isFull()

{

return top == 9;

}

//push value function

void push(int x)

{

if (isFull())

{

cout << "Stack is full! bye bye";

return;

}

arr[top] = x;

top++;

}

//peek value function

void peek(int &x)

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

x = arr[top];

}

//peek value function

void pop()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

top--;

}

//showing stack data

void show()

{

for (int i = 0; i < 10; i++)

{

cout << arr[i] << " ";

}

}

void reverse()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

int temp[10];

for (int i = 0; i < 10; i++)

{

temp[i] = arr[i];

}

for (int i = 0; i < 10; i++)

{

pop();

}

top = 0;

for (int i = 9; i >= 0; i--)

{

arr[top] = temp[i];

top++;

}

}

};

int main()

{

Stack stack;

int n;

while (1)

{

system("cls");

cout << "1. insert value\n2. pop value\n3. peek value\n4. is Empty\n5. is Full\n6. show stack \n7. reverse stack\n8. exit\n";

int num;

cin >> num;

switch (num)

{

case 1:

cout << "\nEnter number: ";

cin >> n;

stack.push(n);

system("pause");

break;

case 2:

stack.pop();

system("pause");

break;

case 3:

stack.peek(n);

cout << "number peek: " << n;

system("pause");

break;

case 4:

stack.isEmpty();

system("pause");

break;

case 5:

stack.isFull();

system("pause");

break;

case 6:

stack.show();

system("pause");

break;

case 7:

stack.reverse();

system("pause");

break;

case 8:

exit(0);

system("pause");

break;

default:

break;

}

}

}

**Part b**

#include<iostream>

#include<string>

using namespace std;

class Stack {

int size=10;

int arr[10] = { 0 };

int top;

public:

//constructor

Stack():top(0){}

//is empty function

bool isEmpty()

{

return top == 0;

}

//is full fuunction

bool isFull()

{

return top == 9;

}

//push value function

void push(int x)

{

if (isFull())

{

cout << "Stack is full! bye bye";

return;

}

arr[top] = x;

top++;

}

//peek value function

void peek(int &x)

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

x = arr[top];

}

//peek value function

void pop()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

top--;

}

//showing stack data

void show()

{

for (int i = 0; i < 10; i++)

{

cout << arr[i] << " ";

}

}

void sort()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

int temp[10];

for (int i = 0; i < 10; i++)

{

temp[i] = arr[i];

}

for (int i = 0; i < 10; i++)

{

pop();

}

top = 0;

//sorting

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

{

for (int j = i+1; j < 9; j++)

{

if (temp[i] < temp[j])

{

int tem = temp[i];

temp[i] = temp[j];

temp[j] = tem;

}

}

}

for (int i = 9; i >= 0; i--)

{

arr[top] = temp[i];

top++;

}

}

};

int main()

{

Stack stack;

int n;

while (1)

{

system("cls");

cout << "1. insert value\n2. pop value\n3. peek value\n4. is Empty\n5. is Full\n6. show stack \n7. sort stack\n8. exit\n";

int num;

cin >> num;

switch (num)

{

case 1:

cout << "\nEnter number: ";

cin >> n;

stack.push(n);

system("pause");

break;

case 2:

stack.pop();

system("pause");

break;

case 3:

stack.peek(n);

cout << "number peek: " << n;

system("pause");

break;

case 4:

stack.isEmpty();

system("pause");

break;

case 5:

stack.isFull();

system("pause");

break;

case 6:

stack.show();

system("pause");

break;

case 7:

stack.sort();

system("pause");

break;

case 8:

exit(0);

system("pause");

break;

default:

break;

}

}

}

**Part C**

#include<iostream>

#include<string>

using namespace std;

class Stack {

int size=10;

int arr[10] = { 0 };

int top;

public:

//constructor

Stack():top(0){}

//is empty function

bool isEmpty()

{

return top == 0;

}

//is full fuunction

bool isFull()

{

return top == 9;

}

//push value function

void push(int x)

{

if (isFull())

{

cout << "Stack is full! bye bye";

return;

}

arr[top] = x;

top++;

}

//peek value function

void peek(int &x)

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

x = arr[top];

}

//peek value function

void pop()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

top--;

}

//showing stack data

void show()

{

for (int i = 0; i < 10; i++)

{

cout << arr[i] << " ";

}

}

//duplicate

void remove()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

int temp[10] = { 0 };

//removing duplicate

bool check[10 ]= { false };

int j = 0;

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

{

if (!check[arr[i]])

{

check[arr[i]] = true;

temp[j] = arr[i];

j++;

}

}

for (int i = 0; i < 10; i++)

{

pop();

}

top = 0;

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

{

arr[top] = temp[i];

top++;

}

}

};

int main()

{

Stack stack;

int n;

while (1)

{

system("cls");

cout << "1. insert value\n2. pop value\n3. peek value\n4. is Empty\n5. is Full\n6. show stack \n7. remove duplicate\n8. exit\n";

int num;

cin >> num;

switch (num)

{

case 1:

cout << "\nEnter number: ";

cin >> n;

stack.push(n);

system("pause");

break;

case 2:

stack.pop();

system("pause");

break;

case 3:

stack.peek(n);

cout << "number peek: " << n;

system("pause");

break;

case 4:

stack.isEmpty();

system("pause");

break;

case 5:

stack.isFull();

system("pause");

break;

case 6:

stack.show();

system("pause");

break;

case 7:

stack.remove();

system("pause");

break;

case 8:

exit(0);

system("pause");

break;

default:

break;

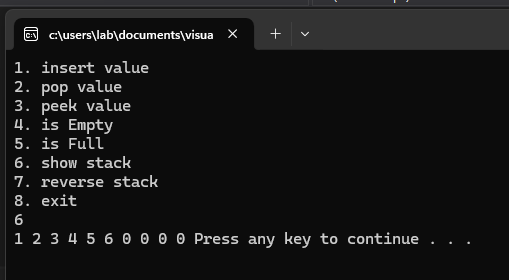
}

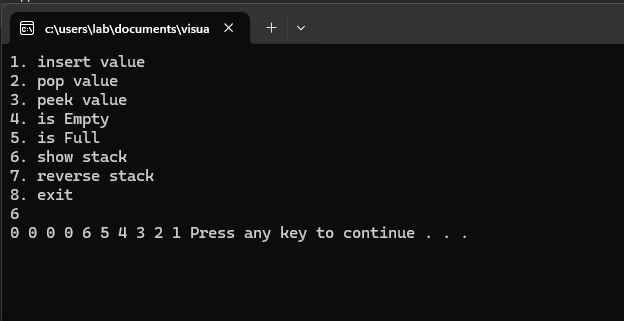
}

}

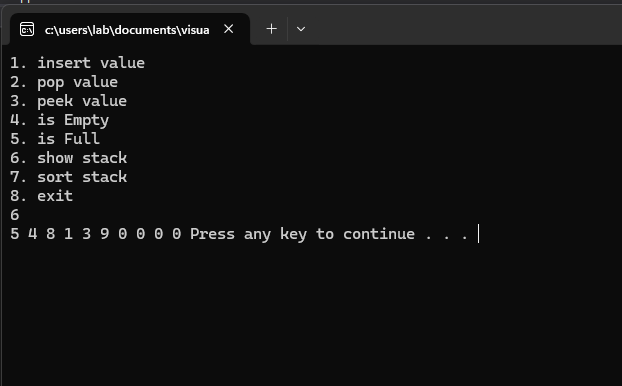
**OUTPUT:**

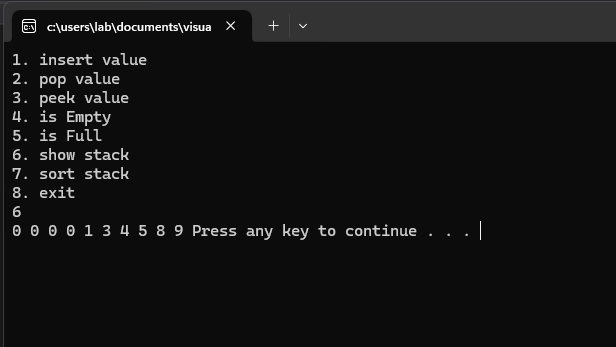
**Part A**



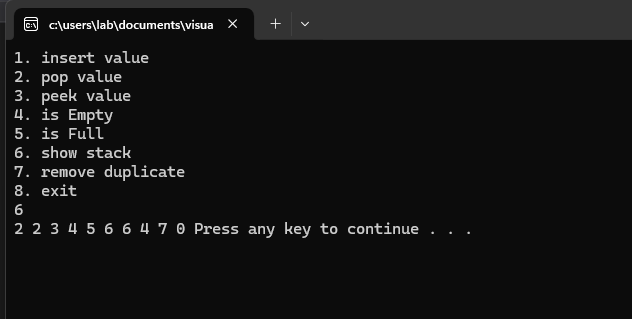


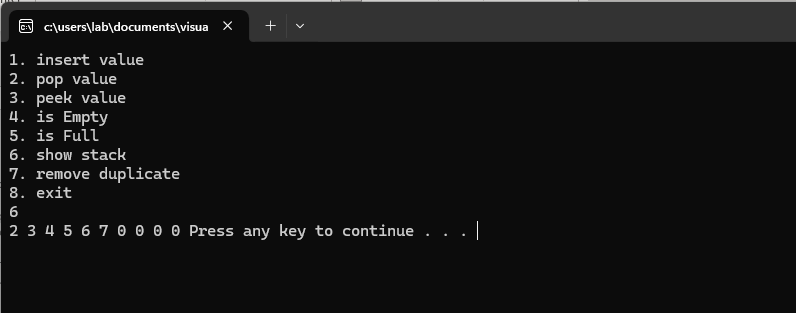
**Part b**

****

****

**Part C**





**TASK3:**

**CODE:**

#include<iostream>

#include<string>

using namespace std;

struct Node {

float cgpa;

int roll;

int marks;

Node\* next;

Node():next(nullptr){}

};

class Linkedlist {

Node\* head = nullptr;

Node\* newnode = nullptr;

Node\* current = nullptr;

int count = 0;

public:

Linkedlist() {}

//destructor

~Linkedlist()

{

current = head;

newnode = nullptr;

while (current != nullptr)

{

newnode = current->next;

delete current;

current = newnode;

}

current = nullptr;

newnode = nullptr;

delete current;

delete newnode;

}

void add()

{

newnode = new Node();

int roll;

cout << "enter roll no: ";

cin >> roll;

current = head;

while (current != nullptr)

{

if (current->roll == roll)

{

cout << "data cannot be entered roll no is same!";

return;

}

current = current->next;

}

newnode->roll = roll;

cout << "enter cgpa: ";

cin >> newnode->cgpa;

cout << "\nenter marks one by one of all subjects: ";

int num, sum = 0;

for (int i = 0; i < 6; i++)

{

cin >> num;

sum = sum + num;

}

newnode->marks = sum;

if (head == nullptr)

{

head = newnode;

}

else

{

current = head;

head = newnode;

head->next = current;

}

}

//print

void print()

{

current = head;

while (current != nullptr)

{

cout << "\n\nRoll No: " << current->roll << "\nCGPA: " << current->cgpa << "\nMarks: " << current->marks;

current = current->next;

}

}

};

class Stack {

Linkedlist list[10];

int size=10;

int top;

public:

//constructor

Stack():top(0){}

//is empty function

bool isEmpty()

{

return top == 0;

}

//is full fuunction

bool isFull()

{

return top == 9;

}

//push value function

void push()

{

if (isFull())

{

cout << "Stack is full! bye bye";

return;

}

list[top].add();

top++;

}

//peek value function

void pop()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

top--;

}

//showing stack data

void show()

{

for (int i = 0; i < 10; i++)

{

list[i].print();

cout << "\n\n";

}

}

};

int main()

{

Stack stack;

int n;

while (1)

{

system("cls");

cout << "1. insert node\n2. pop noden3. is Empty\n4. show stack \n5. exit\n";

int num;

cin >> num;

switch (num)

{

case 1:

stack.push();

system("pause");

break;

case 2:

stack.pop();

system("pause");

break;

case 3:

stack.isEmpty();

system("pause");

break;

case 4:

stack.show();

system("pause");

break;

case 5:

exit(0);

system("pause");

break;

default:

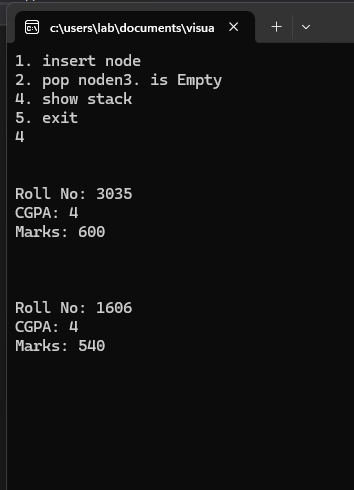
break;

}

}

}

**OUTPUT:**

****

**TASK4:**

**CODE:**

#include<iostream>

#include<string>

using namespace std;

class Stack {

int size = 10;

string arr[10] = { " " };

int top;

public:

//constructor

Stack() :top(0) {}

//is empty function

bool isEmpty()

{

return top == 0;

}

//is full fuunction

bool isFull()

{

return top == 9;

}

//push value function

void push(string x)

{

if (isFull())

{

cout << "Stack is full! bye bye";

return;

}

arr[top] = x;

top++;

}

//peek value function

void current()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

cout << "current element is: ";

cout << arr[top];

}

//peek value function

void pop()

{

if (isEmpty())

{

cout << "Stack is empty! bye bye";

return;

}

top--;

}

//showing stack data

void show()

{

for (int i = 0; i < 10; i++)

{

cout << arr[i] << " ";

cout << endl;

}

}

//clear history

void clear()

{

for (int i = 0; i < 10; i++)

{

pop();

}

top = 0;

}

//forward navigate

void forward()

{

if (arr[top + 1] == " ")

{

cout << " \nforward element is empty!";

}

else

{

top++;

cout << "\nforward elemnt: ";

cout << arr[top];

}

}

//backward navigate

void backward()

{

top--;

cout << "\nbackward elemnt: ";

cout << arr[top];

}

};

int main()

{

Stack stack;

int n;

string num;

while (1)

{

system("cls");

cout << "1. insert URL\n2. pop URL\n3. forward navigate\n4. backward navigate\n5. current elemnt\n6. show stack\n7. clear history \n8. exit\n";

cin >> n;

switch (n)

{

case 1:

cout << "\nEnter URL: ";

cin.ignore();

stack.push(num);

getline(cin, num);

system("pause");

break;

case 2:

stack.pop();

system("pause");

break;

case 3:

stack.forward();

system("pause");

break;

case 4:

stack.backward();

system("pause");

break;

case 5:

stack.current();

system("pause");

break;

case 6:

stack.show();

system("pause");

break;

case 7:

stack.clear();

system("pause");

break;

case 8:

exit(0);

system("pause");

break;

default:

break;

}

}

}

**OUTPUT:**

