A logo with blue and red text

AI-generated content may be incorrect.

**Worksheet – 4**

**Name : Crishtina K.C.**

**Student ID : 23085130**

**Cyber Security And Digital Forensics**

**Task 1: Solve the following programming problems:**

1. STL Container Practice: Write a program using STL containers that: (40 marks)
   1. Uses vector<string> to store names (5 Marks)
   2. Uses map<string, int> to store age against each name (5 Marks)
   3. Implements functions to:
      1. Add new name-age pair (10 marks)
      2. Find all people above certain age (10 marks)
      3. Sort and display names alphabetically (10 marks)

#include <iostream>

#include <vector>

#include <map>

#include <algorithm>

using namespace std;

class PeopleManagement

{

private:

vector<string> names;

map<string, int> ageMap;

public:

void addPerson(const string& name, int age)

{

if (ageMap.find(name) == ageMap.end())

{

names.push\_back(name);

ageMap[name] = age;

cout << " Person Added: " << name << " (" << age << " years old)\n"<<endl;

}

else

{

cout << "Entered Name already exists IN FILE . Updating age for " << name << ".\n"<<endl;

ageMap[name] = age;

}

}

void findPeopleAboveAge(int ageThreshold)

{

cout << "\n THE PEOPLES older than: " << ageThreshold << ":\n"<<endl;

bool found = false;

for (const auto& name : names)

{

if (ageMap[name] > ageThreshold)

{

cout << name << ":: - " << ageMap[name] << " in years\n"<<endl;

found = true;

}

}

if (!found)

{

cout << "No matching people found above age Enterd " << ageThreshold << ".\n";

}

}

void displaySortedNames()

{

vector<string> sortedNames = names;

sort(sortedNames.begin(), sortedNames.end());

cout << "\nNames of peoples in alphabetical order:\n";

for (const auto& name : sortedNames)

{

cout << name << " - " << ageMap[name] << " years\n";

}

}

void displayMenu()

{

cout<<"------------------------------";

cout << "\n Person Manager Menu \n";

cout<<"------------------------------"<<endl;

cout << "1. Add any person\n";

cout << "2. Find people above a age you want: \n";

cout << "3. Display names of peoples alphabetically\n";

cout << "4. Exit program \n";

cout << "Enter your choice: ";

}

void handleMenuChoice(int choice)

{

switch (choice)

{

case 1:

{

string name;

int age;

cout << "Enter name of people : "<<endl;

cin >> name;

cout << "Enter age of people : "<<endl;

cin >> age;

addPerson(name, age);

break;

}

case 2:

{

int ageThreshold;

cout << "Enter age threshold you want : "<<endl;

cin >> ageThreshold;

findPeopleAboveAge(ageThreshold);

break;

}

case 3:

displaySortedNames();

break;

case 4:

cout << "Exiting the program...\n"<<endl;

break;

default:

cout << "Invalid choice. Please make correct choice.\n"<<endl;

}

}

};

int main()

{

PeopleManagement m;

int choice;

do

{

m.displayMenu();

cin >> choice;

cout<<endl;

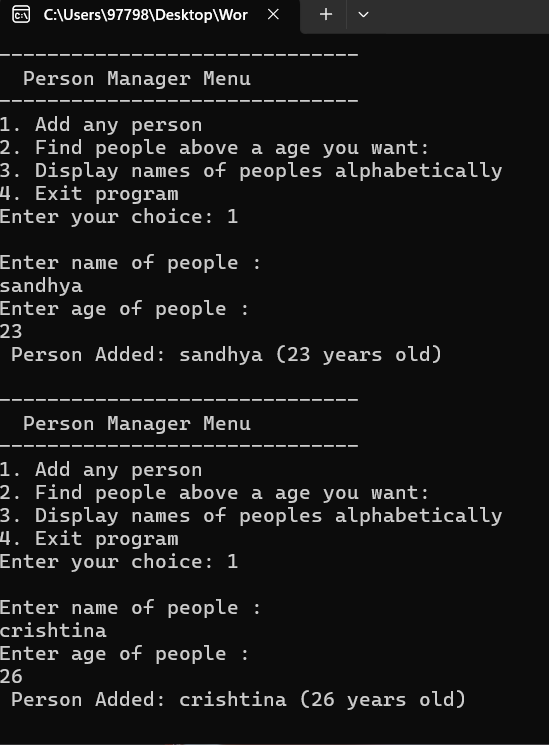
m.handleMenuChoice(choice);

}

while (choice != 4);

return 0;

}



A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Stack Problem: Implement a stack using arrays (not STL) that:
   1. Has basic push and pop operations
   2. Has a function to find middle element
   3. Has a function to reverse only bottom half of stack
   4. Maintain stack size of 10

#include <iostream>

using namespace std;

class Stack

{

private:

int arr[10];

int top;

public:

Stack()

{

top = -1;

}

bool isFull()

{

return top == 9;

}

bool isEmpty()

{

return top == -1;

}

void push(int value)

{

if (isFull())

{

cout << "Stack has Overflow! Cannot push more elements " << value << endl;

return;

}

arr[++top] = value;

cout << "Pushed Number " << value << " into the stack.\n";

}

void pop()

{

if (isEmpty())

{

cout << "Stack Underflow!\n";

return;

}

cout << "Popped " << arr[top--] << " from stack.\n";

}

void display()

{

if (isEmpty())

{

cout << "your Stack is empty.\n";

return;

}

cout << "Showing Stack from top to bottom:\n";

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

{

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

}

cout << endl;

}

void findMiddle()

{

if (isEmpty())

{

cout << "Stack has no any middle numbers: \n";

return;

}

int middleIndex = top / 2;

cout << "The Middle element: " << arr[middleIndex] << endl;

}

void reverseBottomHalf()

{

if (top < 1)

{

cout << "No any elements to reverse bottom half.\n";

return;

}

int n = top + 1;

int mid = n / 2;

for (int i = 0; i < mid / 2; i++)

{

swap(arr[i], arr[mid - 1 - i]);

}

cout << "Bottom half of stack is reversed.\n";

}

};

int main()

{

Stack s;

int choice, value;

do

{

cout<<"------------------------------"<<endl;

cout << "\n==== Stack Menu ====\n";

cout<<"------------------------------"<<endl;

cout << "1. Push\n";

cout << "2. Pop \n";

cout << "3. Display Stack \n";

cout << "4. Find Middle number\n";

cout << "5. Reverse Bottom Half number \n";

cout << "6. Exit Program..\n"<<endl;

cout << "Enter your choice: ";

cin >> choice;

switch (choice)

{

case 1:

cout << "Enter any integer value to push: ";

cin >> value;

s.push(value);

break;

case 2:

s.pop();

break;

case 3:

s.display();

break;

case 4:

s.findMiddle();

break;

case 5:

s.reverseBottomHalf();

break;

case 6:

cout << "Exiting program...\n";

break;

default:

cout << "Invalid choice ! make correct choice.\n";

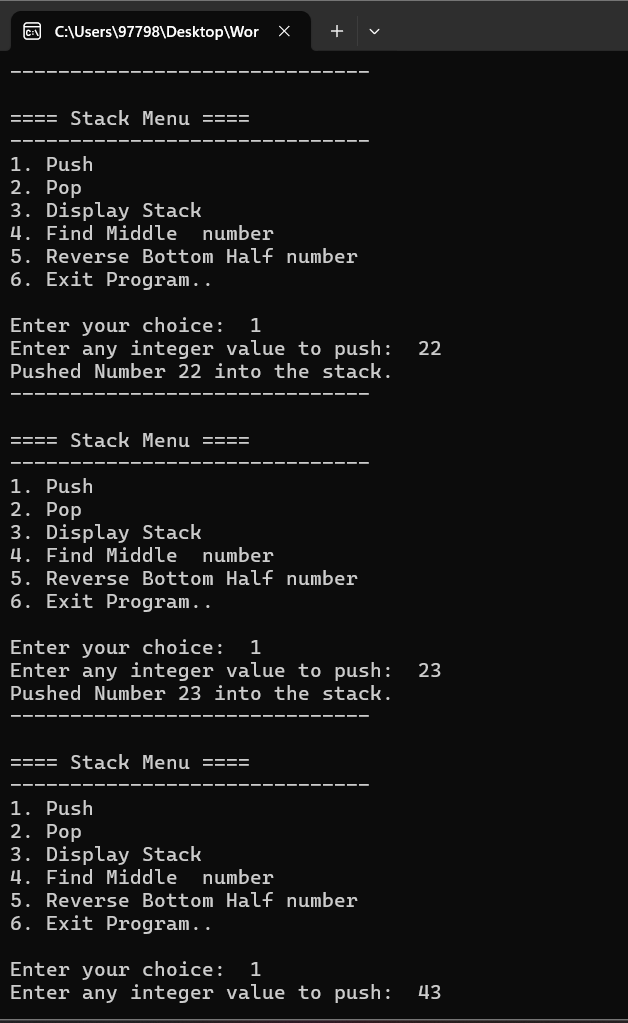
}

}

while (choice != 6);

return 0;

}

A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.

1. Queue Problem: Implement a queue using arrays (not STL) that:
   1. Has basic enqueue and dequeue operations
   2. Has a function to reverse first K elements
   3. Has a function to interleave first half with second half
   4. Handle queue overflow/underflow

#include <iostream>

using namespace std;

#define SIZE 10

class Queue

{

private:

int arr[SIZE];

int front, rear;

public:

Queue()

{

front = rear = -1;

}

bool isEmpty()

{

return front == -1;

}

bool isFull()

{

return (rear + 1) % SIZE == front;

}

void enqueue(int value)

{

if (isFull())

{

cout << "Queue Overflow! Cannot enqueue " << value << endl;

return;

}

if (isEmpty())

{

front = rear = 0;

}

else

{

rear = (rear + 1) % SIZE;

}

arr[rear] = value;

cout << "Enqueued: " << value << endl;

}

void dequeue()

{

if (isEmpty())

{

cout << "Queue Underflow!\n";

return;

}

cout << "Dequeued: " << arr[front] << endl;

if (front == rear)

{

front = rear = -1;

}

else

{

front = (front + 1) % SIZE;

}

}

void display()

{

if (isEmpty())

{

cout << "Queue is empty.\n";

return;

}

cout << "Queue: ";

int i = front;

while (true)

{

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

if (i == rear) break;

i = (i + 1) % SIZE;

}

cout << endl;

}

int size()

{

if (isEmpty()) return 0;

if (rear >= front) return rear - front + 1;

return SIZE - front + rear + 1;

}

void reverseFirstK(int k)

{

if (k > size() || k <= 0)

{

cout << "Invalid value of K.\n";

return;

}

int temp[SIZE];

int count = 0;

int i = front;

for (int j = k - 1; j >= 0; --j)

{

temp[j] = arr[i];

i = (i + 1) % SIZE;

count++;

}

i = front;

for (int j = 0; j < k; ++j)

{

arr[i] = temp[j];

i = (i + 1) % SIZE;

}

cout << "Reversed first " << k << " elements.\n";

}

void interleave()

{

int n = size();

if (n % 2 != 0)

{

cout << "Queue size must be even to interleave.\n";

return;

}

int firstHalf[n/2], secondHalf[n/2];

int i = front;

for (int j = 0; j < n / 2; ++j)

{

firstHalf[j] = arr[i];

i = (i + 1) % SIZE;

}

for (int j = 0; j < n / 2; ++j)

{

secondHalf[j] = arr[i];

i = (i + 1) % SIZE;

}

i = front;

for (int j = 0; j < n / 2; ++j)

{

arr[i] = firstHalf[j];

i = (i + 1) % SIZE;

arr[i] = secondHalf[j];

i = (i + 1) % SIZE;

}

cout << "Interleaved first half with second half.\n";

}

};

int main()

{

Queue q;

int choice, value, k;

do

{

cout << "\n=== Queue Menu ===\n";

cout << "1. Enqueue\n";

cout << "2. Dequeue\n";

cout << "3. Display\n";

cout << "4. Reverse First K Elements\n";

cout << "5. Interleave Queue\n";

cout << "6. Exit\n";

cout << "Enter your choice: "<<endl;

cin >> choice;

switch (choice)

{

case 1:

cout << "Enter value to enqueue: ";

cin >> value;

q.enqueue(value);

break;

case 2:

q.dequeue();

break;

case 3:

q.display();

break;

case 4:

cout << "Enter value of K: ";

cin >> k;

q.reverseFirstK(k);

break;

case 5:

q.interleave();

break;

case 6:

cout << "Exiting program.\n";

break;

default:

cout << "Invalid choice! Try again.\n";

}

} while (choice != 6);

return 0;

}

A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.

1. Linked List Problem: Create a singly linked list (not STL) that:
   1. Has functions to insert at start/end/position
   2. Has a function to detect and remove loops
   3. Has a function to find nth node from end
   4. Has a function to reverse list in groups of K nodes

#include <iostream>

using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int val)

{

data = val;

next = nullptr;

}

};

class LinkedList

{

private:

Node\* head;

public:

LinkedList()

{

head = nullptr;

}

void insertAtStart(int val)

{

Node\* newNode = new Node(val);

newNode->next = head;

head = newNode;

}

void insertAtEnd(int val)

{

Node\* newNode = new Node(val);

if (!head)

{

head = newNode;

return;

}

Node\* temp = head;

while (temp->next)

temp = temp->next;

temp->next = newNode;

}

void insertAtPosition(int pos, int val)

{

if (pos < 1)

{

cout << "Invalid position selected!\n";

return;

}

if (pos == 1)

{

insertAtStart(val);

return;

}

Node\* newNode = new Node(val);

Node\* temp = head;

for (int i = 1; temp != nullptr && i < pos - 1; ++i)

temp = temp->next;

if (!temp)

{

cout << "Position are out of bounds.\n";

return;

}

newNode->next = temp->next;

temp->next = newNode;

}

void detectAndRemoveLoop()

{

Node \*slow = head, \*fast = head;

bool loopFound = false;

while (fast && fast->next)

{

slow = slow->next;

fast = fast->next->next;

if (slow == fast)

{

loopFound = true;

break;

}

}

if (!loopFound)

{

cout << "No loops are detected.\n";

return;

}

slow = head;

Node\* prev = nullptr;

while (slow != fast)

{

prev = fast;

slow = slow->next;

fast = fast->next;

}

if (prev)

prev->next = nullptr;

cout << "Loop has been detected and removed.\n";

}

void findNthFromEnd(int n)

{

Node \*mainPtr = head, \*refPtr = head;

int count = 0;

while (count < n)

{

if (!refPtr)

{

cout << "The given list is shorter than " << n << " nodes.\n";

return;

}

refPtr = refPtr->next;

count++;

}

while (refPtr)

{

mainPtr = mainPtr->next;

refPtr = refPtr->next;

}

cout << "The " << n << "th number node from end is: " << mainPtr->data << endl;

}

Node\* reverseInGroups(Node\* node, int k)

{

Node\* prev = nullptr;

Node\* current = node;

Node\* next = nullptr;

int count = 0;

while (current && count < k)

{

next = current->next;

current->next = prev;

prev = current;

current = next;

count++;

}

if (next)

node->next = reverseInGroups(next, k);

return prev;

}

void reverseInGroups(int k)

{

head = reverseInGroups(head, k);

cout << "List reversed in groups of: " << k << ".\n";

}

void display()

{

Node\* temp = head;

cout << "Linked List or Data : ";

while (temp)

{

cout << temp->data << " -> ";

temp = temp->next;

}

cout << "\_\_\_\_\_\_NULL\_\_\_\_\_\n";

}

void createLoop(int pos)

{

if (pos <= 0) return;

Node\* loopNode = head;

for (int i = 1; i < pos && loopNode; ++i)

loopNode = loopNode->next;

Node\* temp = head;

while (temp->next)

temp = temp->next;

temp->next = loopNode;

cout << "the Loop is created at position: " << pos << ".\n";

}

};

int main()

{

LinkedList list;

int choice, val, pos, k;

do

{

cout << "\n=== Linked List Menu ===\n";

cout << "1. Insert at Start\n";

cout << "2. Insert at End\n";

cout << "3. Insert at Position\n";

cout << "4. Display List\n";

cout << "5. Find Nth Node from End\n";

cout << "6. Reverse in Groups of K\n";

cout << "7. Create Loop (Test)\n";

cout << "8. Detect & Remove Loop\n";

cout << "9. Exit\n";

cout << "Enter choice: ";

cin >> choice;

switch (choice)

{

case 1:

cout << "Enter the start value: ";

cin >> val;

list.insertAtStart(val);

break;

case 2:

cout << "Enter your end value: ";

cin >> val;

list.insertAtEnd(val);

break;

case 3:

cout << "Enter the position and value: ";

cin >> pos >> val;

list.insertAtPosition(pos, val);

break;

case 4:

list.display();

break;

case 5:

cout << "Enter Value of N: ";

cin >> pos;

list.findNthFromEnd(pos);

break;

case 6:

cout << "Enter value of K: ";

cin >> k;

list.reverseInGroups(k);

break;

case 7:

cout << "Enter any position to set loop back to: ";

cin >> pos;

list.createLoop(pos);

break;

case 8:

list.detectAndRemoveLoop();

break;

case 9:

cout << "Exiting the program...\n";

break;

default:

cout << "Invalid choice!! select correct number.\n";

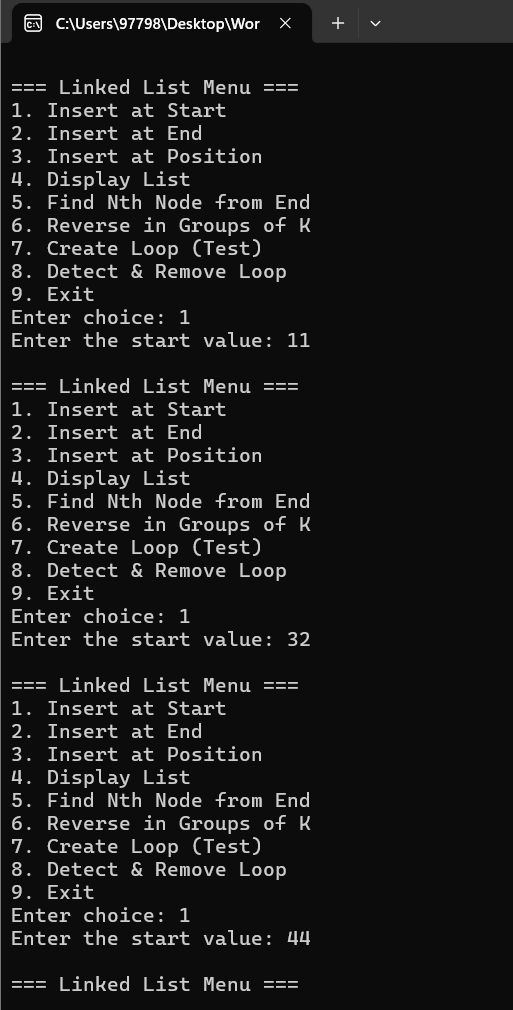
}

}

while (choice != 9);

return 0;

}

A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.