

**National University of Computer and Emerging**

**Sciences**

**Chiniot-Faisalabad Campus BS (Artificial Intelligence)**

|  |  |
| --- | --- |
| **Name** | **M.Abdul Hanan** |
| **Reg.NO.** | **22F-3104** |
| **Section** | **BS(Ai)-3A1** |
| **Course** | **Data Structures** |
| **Department** | **CS Department** |
| **Lab** | **#8** |

**Task no 1:**

#include<iostream>

using namespace std;

int binary\_search(int\* arr, int target, int left, int right) {

if (left <= right) {

int mid = left + (right - left) / 2;

if (arr[mid] == target) {

// Target found at index 'mid'.

return mid;

}

else if (arr[mid] < target) {// Target is in the right half of the current search range

return binary\_search(arr, target, mid + 1, right);

}

else {// Target is in the left half of the current search range

return binary\_search(arr, target, left, mid - 1);

}

}

// Target not found in the array

return -1;

}

int main() {

int\* arr;

int size;

cout << "Enter size of the array in sorted order: ";

cin >> size;

arr = new int[size];

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

cout << "Enter value at index " << i << " : ";

cin >> arr[i];

}

int target;

cout << "Enter a target element: ";

cin >> target;

int result = binary\_search(arr, target, 0, size - 1);

if (result != -1) {

cout << "Target found at index " << result << endl;

}

else {

cout << "Target not found in the array" << endl;

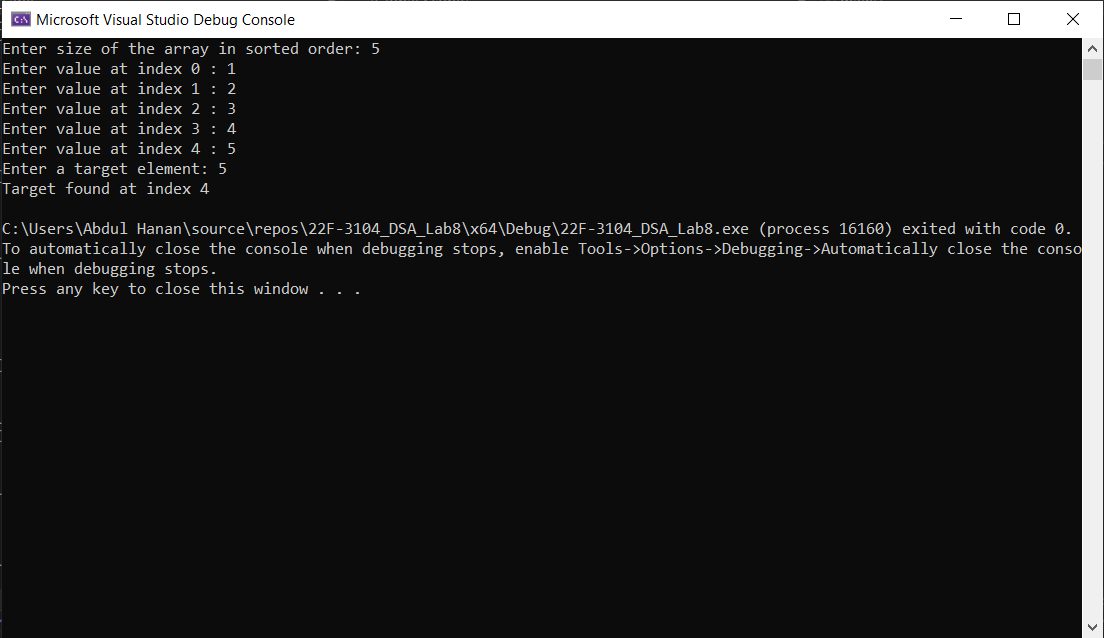
}

delete[] arr;

return 0;

}

**Screen Shot:**



**Task no 2:**

#include <iostream>

using namespace std;

struct Node {

int data;

Node\* next;

Node(int val) {

data = val;

next = nullptr;

}

};

// Function to insert a new node at the beginning of the linked list

int insert(Node\*& head, int val) {

Node\* nn = new Node(val);

Node\* temp = head;

if (temp == nullptr) {

nn->data = val;

nn->next = head;

head = nn;

}

else {

return insert(temp->next, val);

}

}

// Function to reverse a linked list

Node\* reverseLinkedList(Node\* current, Node\* previous = nullptr) {

if (current == nullptr) {

return previous;

}

Node\* next = current->next;

current->next = previous;

return reverseLinkedList(next, current);

}

// Function to print the linked list

void printLinkedList(Node\* head) {

while (head != nullptr) {

cout << head->data<<" ";

head = head->next;

}

}

int main() {

Node\* head = nullptr;

int num; int temp = 0;

// Sentinel loop to keep the program running

while (true) {

cout << "Enter number to insert in the list or press -1 to Exit :";

cin >> num;

temp = num;

if (temp == -1) {

break;

}

insert(head, num);

}

cout << "Original Linked List: ";

printLinkedList(head);

// Reverse the linked list

head = reverseLinkedList(head);

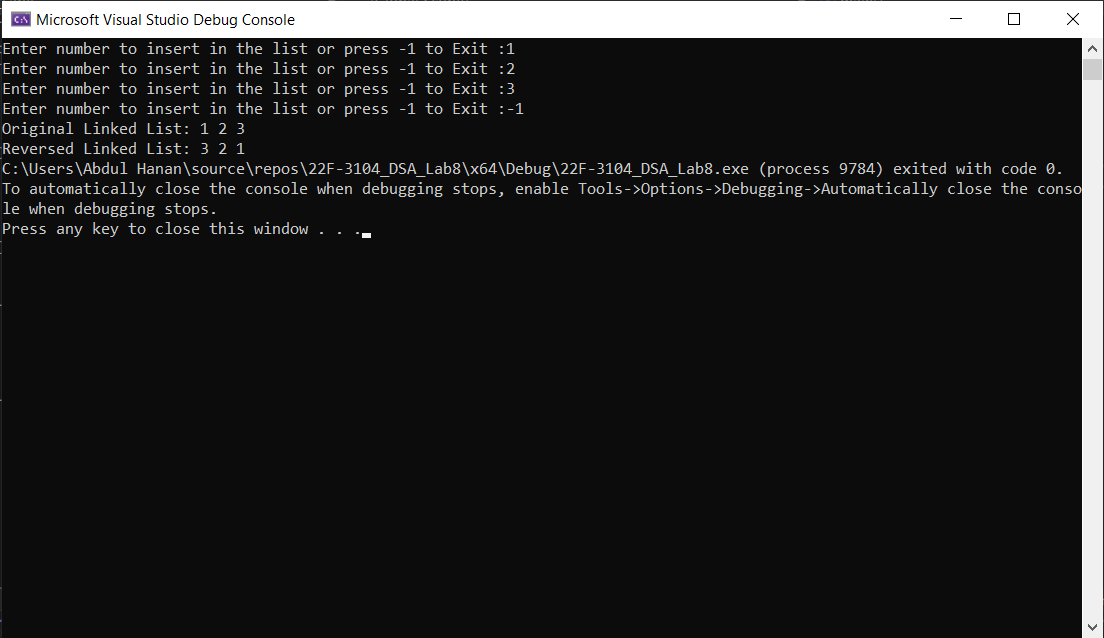
cout << "\nReversed Linked List: ";

printLinkedList(head);

return 0;

}

**Screen Shot:**

****

**Task no 3:**

#include <iostream>

using namespace std;

string decimalToOctal(int decimal, int base) {

if (decimal == 0) {

return "";

}

int remain = decimal % base;

if (remain < 10) {

return decimalToOctal(decimal / base, base) + char('0' + remain);

}

else {

return decimalToOctal(decimal / base, base) + char('A' + remain - 10);

}

}

int main() {

int decimal\_Num, base\_num;

cout << "Enter a decimal number: ";

cin >> decimal\_Num;

cout << "Enter 8 for octal or 16 for hexadecimal: ";

cin >> base\_num;

if (base\_num == 8) {

cout << decimal\_Num << " in octal is: " <<decimalToOctal(decimal\_Num, 8) << endl;

}

else if (base\_num == 16) {

cout << decimal\_Num << " in hexadecimal is:" <<decimalToOctal(decimal\_Num, 16) << endl;

}

else {

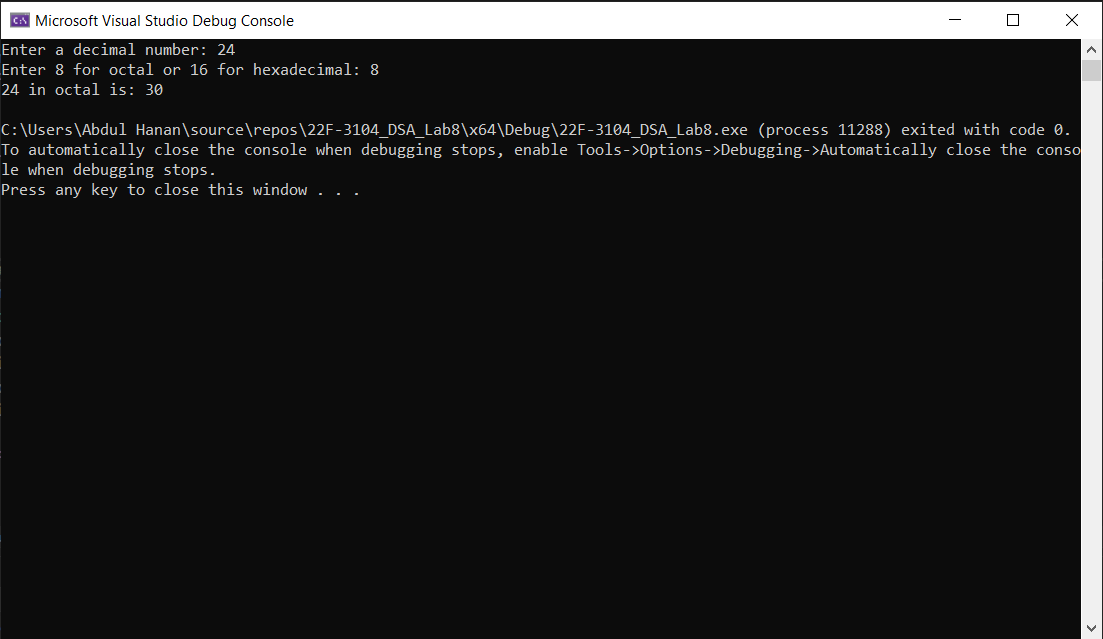
cout << "Invalid base number" << endl;

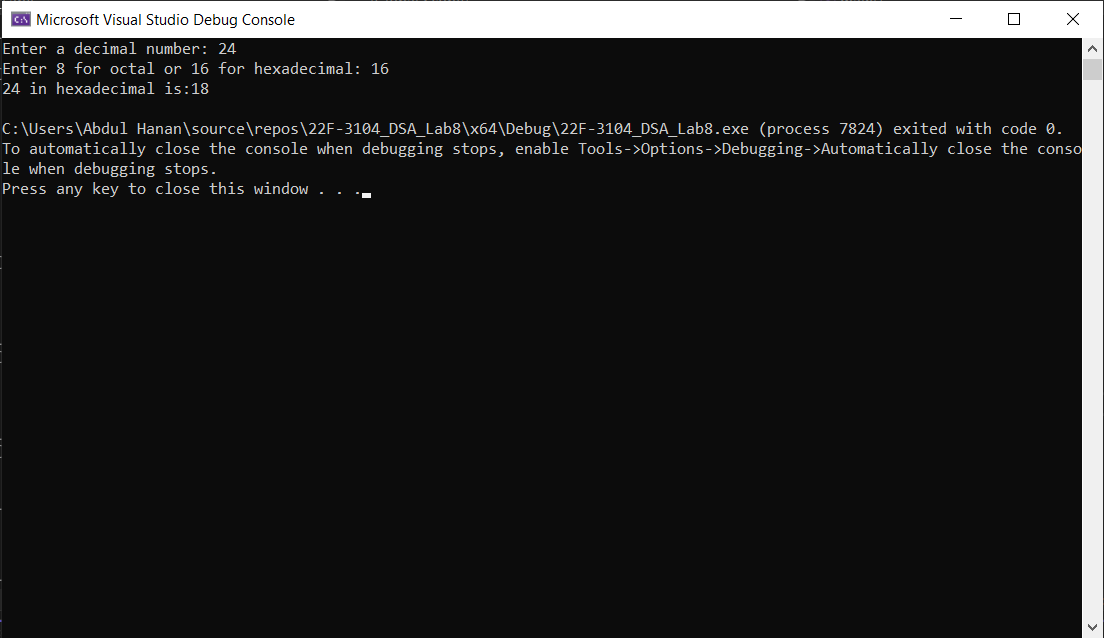
}

return 0;

}

**Screen Shot:**

****

****