**DATA STRUCTURES LAB**

**LAB TASK – 4**

**PROGRAM – 1**

**#include <stdio.h>**

**#define size 10**

**void push();**

**void pop();**

**void display();**

**void peek();**

**int stack[size];**

**int top=-1;**

**void main()**

**{**

**int value,choice;**

**while(1)**

**{**

**printf("\n\nAVAILABLE CHOICES ARE :\n");**

**printf("1.push\n2.pop\n3.peek\n4.display\n5.exit");**

**printf("\n\nenter your choice:");**

**scanf("%d",&choice);**

**switch(choice)**

**{**

**case 1: printf("Enter the value to be inserted: ");**

**scanf("%d",&value);**

**push(value);**

**break;**

**case 2:**

**pop();**

**break;**

**case 3:**

**peek();**

**break;**

**case 4:**

**display();**

**break;**

**case 5:**

**exit(0);**

**default:**

**printf("\ninput error");**

**}**

**}**

**}**

**void push(int value)**

**{**

**if(top == size-1)**

**printf("\nOverflow");**

**else**

**{**

**top++;**

**stack[top] = value;**

**printf("\nInserted successfully");**

**}**

**}**

**void pop()**

**{**

**if(top == -1)**

**printf("\nUnderflow");**

**else**

**{**

**printf("\nDeleted successfully : %d", stack[top]);**

**top--;**

**}**

**}**

**void peek()**

**{**

**if(top == -1)**

**printf("\nUnderflow");**

**else**

**{**

**printf("\nStack top most element is: %d\n",stack[top]);**

**}**

**}**

**void display()**

**{**

**if(top == -1)**

**printf("\nUnderflow");**

**else**

**{**

**printf("\nElements in the stack are:\n");**

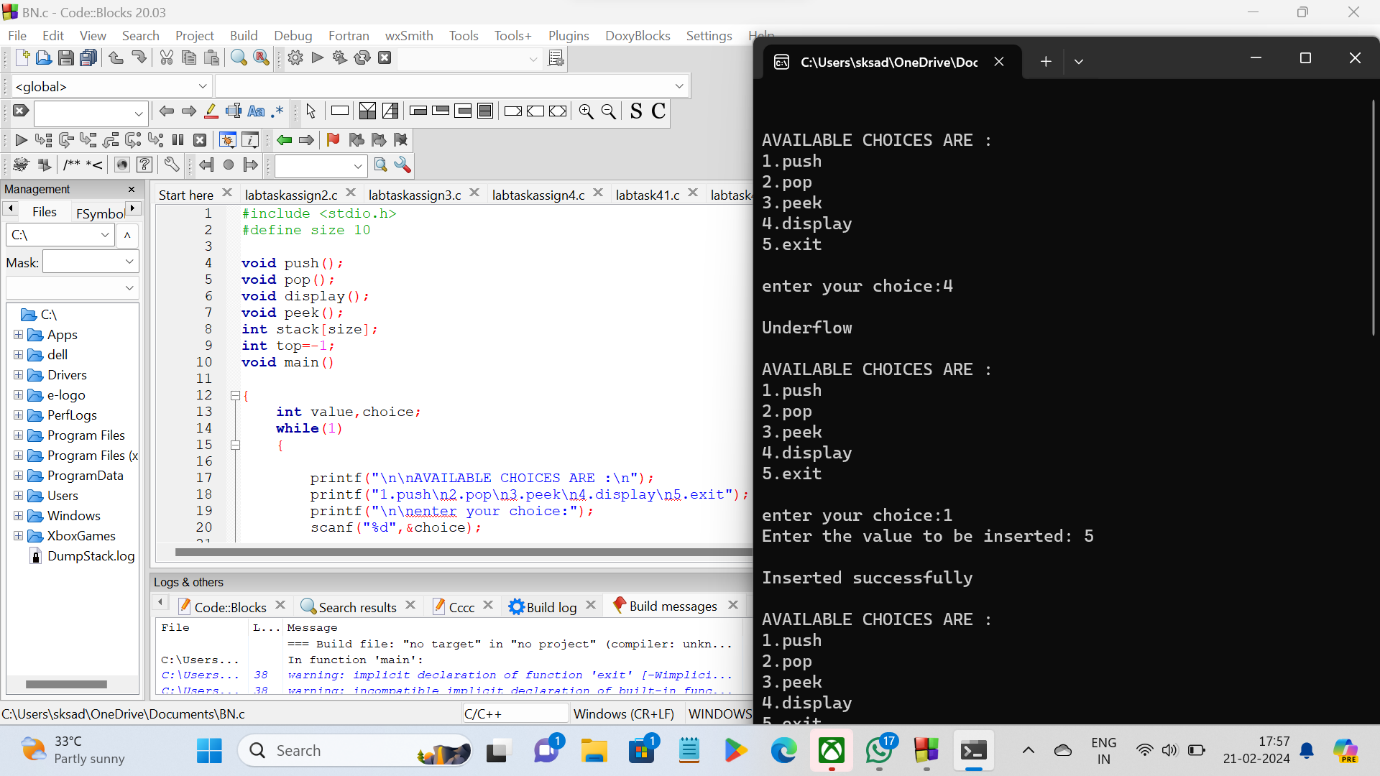
**for(int i=top; i>=0; i--)**

**printf("%d\n",stack[i]);**

**}**

**}**

**OUTPUT :**

****

**PROGRAM – 2**

**#include <stdio.h>**

**char stack[50];**

**int top = -1;**

**void push(char x)**

**{**

**stack[++top] = x;**

**}**

**char pop()**

**{**

**if (top == -1)**

**return -1;**

**else**

**return stack[top--];**

**}**

**int alphanum(char c)**

**{**

**return (c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') || (c >= '0' && c <= '9');**

**}**

**int precedence(char k)**

**{**

**if (k == '(')**

**return 0;**

**if (k == '+' || k == '-')**

**return 1;**

**if (k == '\*' || k == '/')**

**return 2;**

**return 0;**

**}**

**int main()**

**{**

**char infix[50];**

**char k;**

**printf("Enter the infix expression : ");**

**scanf("%s", infix);**

**printf("\n");**

**int i = 0;**

**while (infix[i] != '\0')**

**{**

**if (alphanum(infix[i]))**

**printf("%c ", infix[i]);**

**else if (infix[i] == '(')**

**push(infix[i]);**

**else if (infix[i] == ')')**

**{**

**while ((k = pop()) != '(')**

**printf("%c ", k);**

**}**

**else**

**{**

**while (precedence(stack[top]) >= precedence(infix[i]))**

**printf("%c ", pop());**

**push(infix[i]);**

**}**

**i++;**

**}**

**while (top != -1)**

**{**

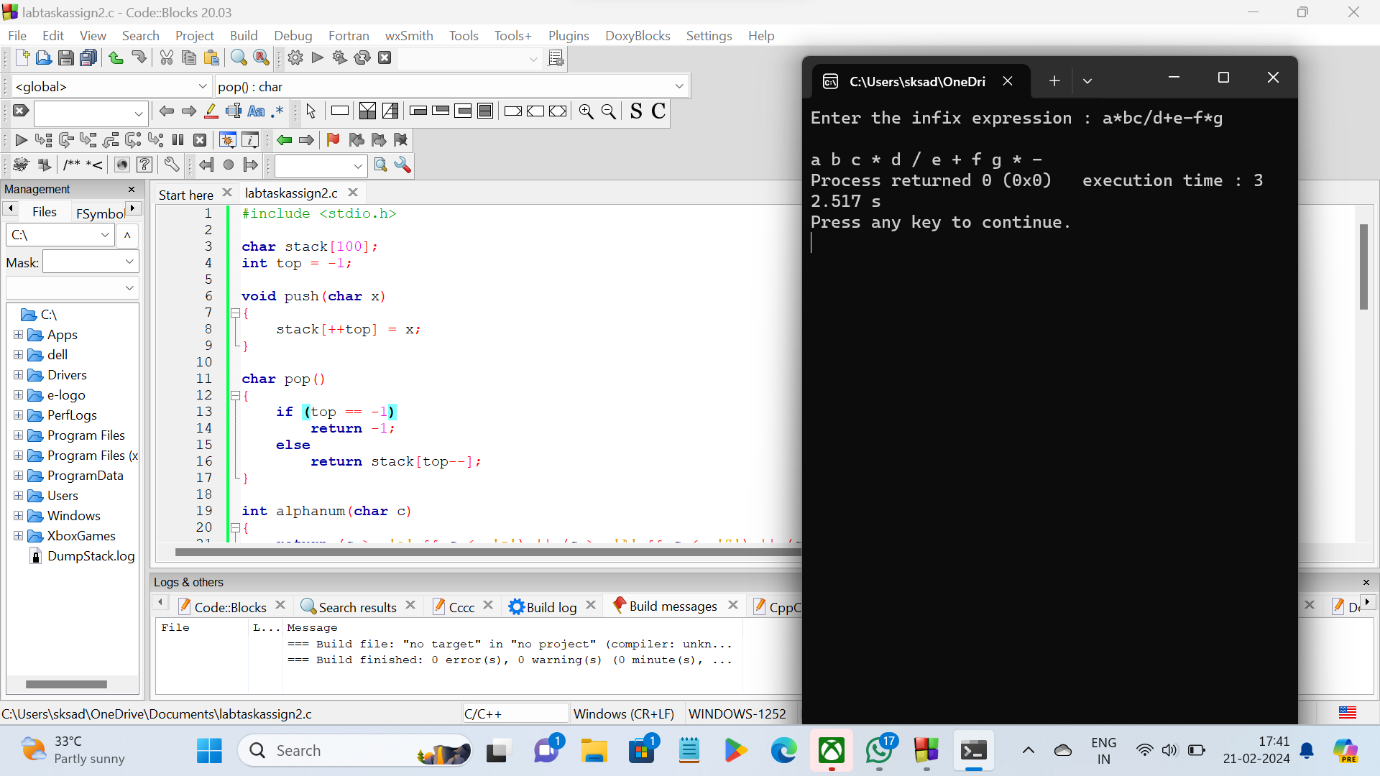
**printf("%c ", pop());**

**}**

**return 0;**

**}**

**OUTPUT :**

****

**PROGRAM - 3**

**#include <stdio.h>**

**char stack[50];**

**int top = -1;**

**void push(char x)**

**{**

**stack[++top] = x;**

**}**

**char pop()**

**{**

**if (top == -1)**

**return -1;**

**else**

**return stack[top--];**

**}**

**int alphanum(char c)**

**{**

**return (c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') || (c >= '0' && c <= '9');**

**}**

**int precedence(char k)**

**{**

**if (k == '(')**

**return 0;**

**if (k == '+' || k == '-')**

**return 1;**

**if (k == '\*' || k == '/')**

**return 2;**

**return 0;**

**}**

**int operand(char c)**

**{**

**return (c >= '0' && c <= '9');**

**}**

**int postfix(char \*result)**

**{**

**int i, operation1, operation2;**

**for (i = 0; result[i] != '\0'; i++)**

**{**

**if (operand(result[i]))**

**{**

**push(result[i] - '0');**

**}**

**else**

**{**

**operation2 = pop();**

**if (operation2 == -1)**

**{**

**printf("Error: Invalid expression\n");**

**return -1;**

**}**

**operation1 = pop();**

**if (operation1 == -1)**

**{**

**printf("Error: Invalid expression\n");**

**return -1;**

**}**

**switch (result[i])**

**{**

**case '+':**

**push(operation1 + operation2);**

**break;**

**case '-':**

**push(operation1 - operation2);**

**break;**

**case '\*':**

**push(operation1 \* operation2);**

**break;**

**case '/':**

**push(operation1 / operation2);**

**break;**

**}**

**}**

**}**

**return stack[top];**

**}**

**int main()**

**{**

**char infix[50];**

**char k;**

**printf("Enter the infix expression : ");**

**scanf("%s", infix);**

**printf("\n");**

**int i = 0;**

**char result[50];**

**int j = 0;**

**while (infix[i] != '\0')**

**{**

**if (alphanum(infix[i]))**

**{**

**result[j++] = infix[i];**

**}**

**else if (infix[i] == '(')**

**{**

**push(infix[i]);**

**}**

**else if (infix[i] == ')')**

**{**

**while ((k = pop()) != '(')**

**{**

**result[j++] = k;**

**}**

**}**

**else**

**{**

**while (precedence(stack[top]) >= precedence(infix[i]))**

**{**

**result[j++] = pop();**

**}**

**push(infix[i]);**

**}**

**i++;**

**}**

**while (top != -1)**

**{**

**result[j++]=pop();**

**}**

**result[j]='\0';**

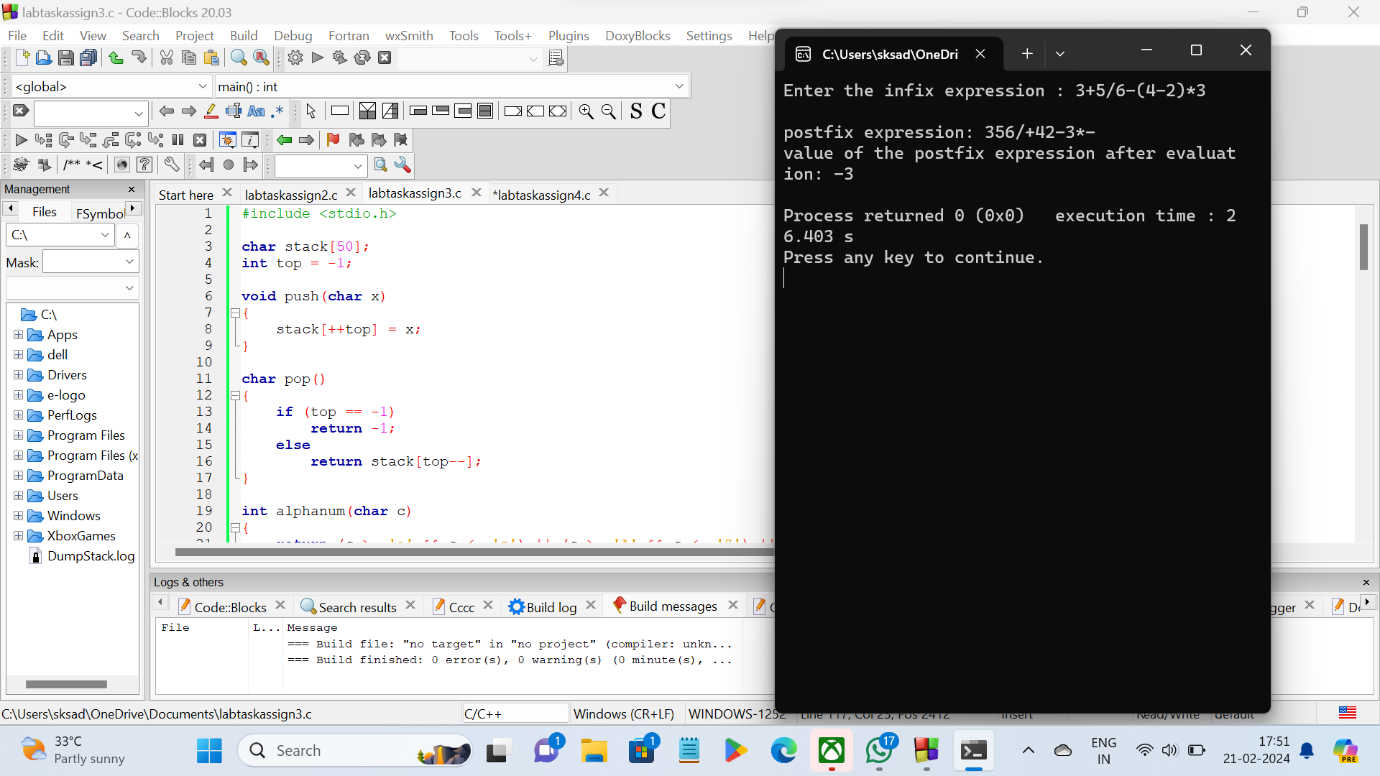
**printf("postfix expression: %s\n",result);**

**printf("value of the postfix expression after evaluation: %d\n",postfix(result));**

**return 0;**

**}**

**OUTPUT :**

****

**PROGRAM - 4**

**#include <stdio.h>**

**void movedisk(char from, char to, int disk)**

**{**

**printf("\nMove disk %d from %c to %c", disk, from, to);**

**}**

**void tower(int num, char source, char dest, char temp)**

**{**

**if (num == 0)**

**return;**

**tower(num - 1, source, temp, dest);**

**movedisk(source, dest, num);**

**tower(num - 1, temp, dest, source);**

**}**

**int main()**

**{**

**int num;**

**printf("Enter number of disks: ");**

**scanf("%d", &num);**

**if (num <= 0)**

**{**

**printf("enter valid number of disks");**

**return 0;**

**}**

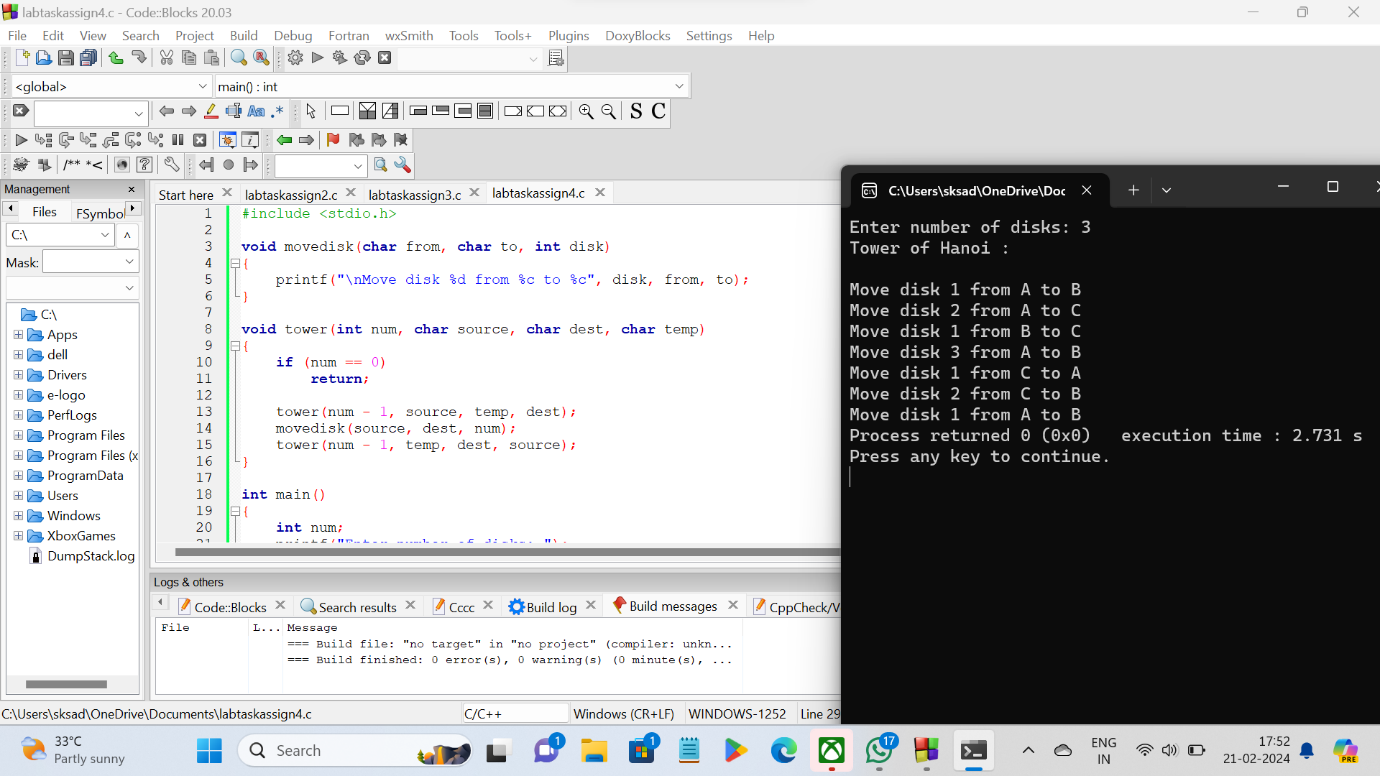
**printf("Tower of Hanoi :\n");**

**tower(num, 'A', 'B', 'C');**

**return 0;**

**}**

**OUTPUT :**

****

**\*\*\*\*\*THE END\*\*\*\*\***