RAJIV GANDHI
INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Data structure Lab manual

2023-2024

Department of CS & E

**RAJIV GANDHI INSTITUTE OF TECHNOLOGY
CHOLANAYAKANAHALLI, BENGALURU-560032**

Course Code	**BCSL305**	**CIE Marks**	50
Number of Contact Hours/Week	0:0:2	**SEE Marks**	50
Total Number of Lab Contact Hours	28	**Exam Hours**	03
Credits – 1			

CONTENTS

4.	4. Design, Develop and Implement a Program in C for converting an Infix Expression to Postfix Expression. Program should support for both parenthesized and free parenthesized expressions withthe operators: +, -, *, /, %(Remainder), ^ (Power) and alphanumeric operands	14
5.	5. Design, Develop and Implement a Program in C for the following Stack Applications a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^ b.Solving Tower of Hanoi problem with n disks. a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^ b. Solving Tower of Hanoi problem with n disks.	18
6.	6. Design, Develop and Implement a menu driven Program in C for the following operations on Circular QUEUE of Characters (Array Implementation of Queue with maximum size MAX) a. Insert an Element on to Circular QUEUE b. Delete an Element from Circular QUEUE c. Demonstrate Overflow and Underflow situations on Circular QUEUE d. Display the status of Circular QUEUE e. Exit Support the program with appropriate functions for each of the above operations	22
7.	7. Design, Develop and Implement a menu driven Program in C for the following operations on Singly Linked List (SLL) of Student Data with the fields: USN, Name, Branch, Sem, PhNo a. Create a SLL of N Students Data by using front insertion. b. Display the status of SLL and count the number of nodes in it c. Perform Insertion and Deletion at End of SLL d. Perform Insertion and Deletion at Front of SLL e. Demonstrate how this SLL can be used as STACK and QUEUE f. Exit	29
8.	8.Design, Develop and Implement a menu driven Program in C for the following operations on Doubly Linked List (DLL) of Employee Data with the fields: SSN, Name, Dept, Designation, Sal, PhNo a. Create a DLL of N Employees Data by using end insertion. b. Display the status of DLL and count the number of nodes in it	38

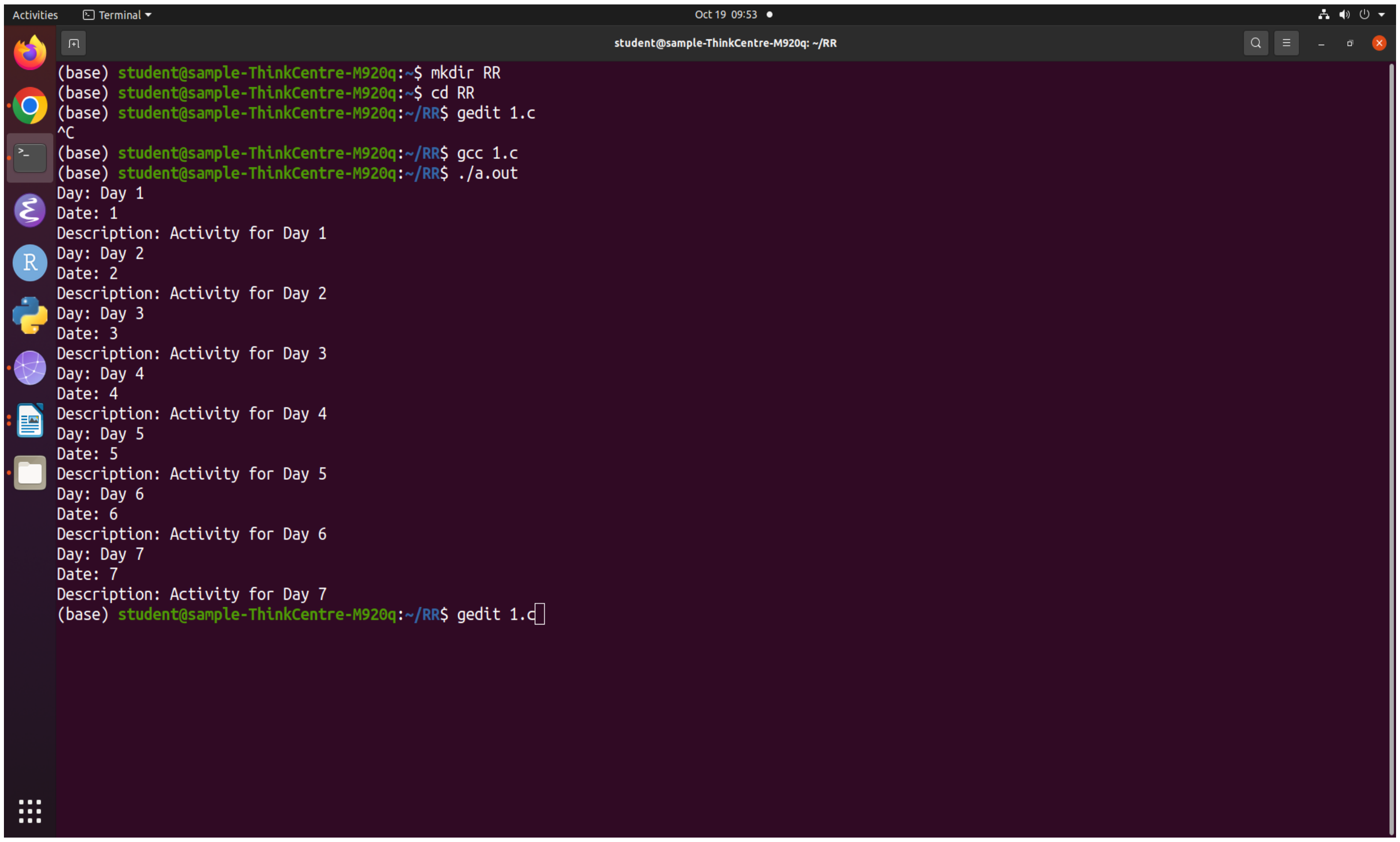
	c. Perform Insertion and Deletion at End of DLL d. Perform Insertion and Deletion at Front of DLL e. Demonstrate how this DLL can be used as Double Ended Queue f. Exit	
9.	9. Design, Develop and Implement a Program in C for the following operations on Singly Circular Linked List (SCLL) with header nodes a. Represent and Evaluate a Polynomial P(x,y,z) = 6x2y2z-4yz5+3x3yz+2xy5z-2xyz3 b. Find the sum of two polynomials POLY1(x,y,z) and POLY2(x,y,z) and store the result in POLYSUM(x,y,z) Support the program with appropriate functions for each of the above operations	46
10.	10. Design, Develop and Implement a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers . a. Create a BST of N Integers: 6, 9, 5, 2, 8, 15, 24, 14, 7, 8, 5, 2 b. Traverse the BST in Inorder, Preorder and Post Order c. Search the BST for a given element (KEY) and report the appropriate message d. Exit.	50
11.	11.Design, Develop and Implement a Program in C for the following operations on Graph(G) of Cities a. Create a Graph of N cities using Adjacency Matrix. b. Print all the nodes reachable from a given starting node in a digraph using BFS/DFS method	56
12.	12. Given a File of N employee records with a set K of Keys (4-digit) which uniquely determine the records in file F. Assume that file F is maintained in memory by a Hash Table (HT) of m memory locations with L as the set of memory addresses (2-digit) of locations in HT. Let the keys in K and addresses in L are Integers. Design and develop a Program in C that uses Hash function H: K --> L as H(K)=K mod m (remainder method), and implement hashing technique to map a given key K to the address space L. Resolve the collision (if any) using linear probing.	59

1. Develop a Program in C for the following:
a. Declare a calendar as an array of 7 elements (A dynamically Created array) to represent 7 days of a week. Each Element of the array is a structure having three fields. The first field is the name of the Day (A dynamically allocated String), The second field is the date of the Day (A integer), the third field is the description of the activity for a particular day (A dynamically allocated String).
b. Write functions create(), read() and display(); to create the calendar, to read the data from
the keyboard and to print weeks activity details report on screen.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Day
{
   char* name;
   int date;
   char* description;
};
int main()
{
   struct Day calendar[7];
   for (int i = 0; i < 7; i++)
         {
                calendar[i].name = (char*)malloc(sizeof(char) * 20);
                calendar[i].description = (char*)malloc(sizeof(char) * 100);
                sprintf(calendar[i].name, "Day %d", i + 1);
                calendar[i].date = i + 1; // Date 1, 2, ...
                sprintf(calendar[i].description, "Activity for Day %d", i + 1);
          }
         for (int i = 0; i < 7; i++)
         {
                printf("Day: %s\n", calendar[i].name);
                printf("Date: %d\n", calendar[i].date);
                printf("Description: %s\n", calendar[i].description);
                free(calendar[i].name);
                free(calendar[i].description);
         }
         return 0;
```

}

Output:



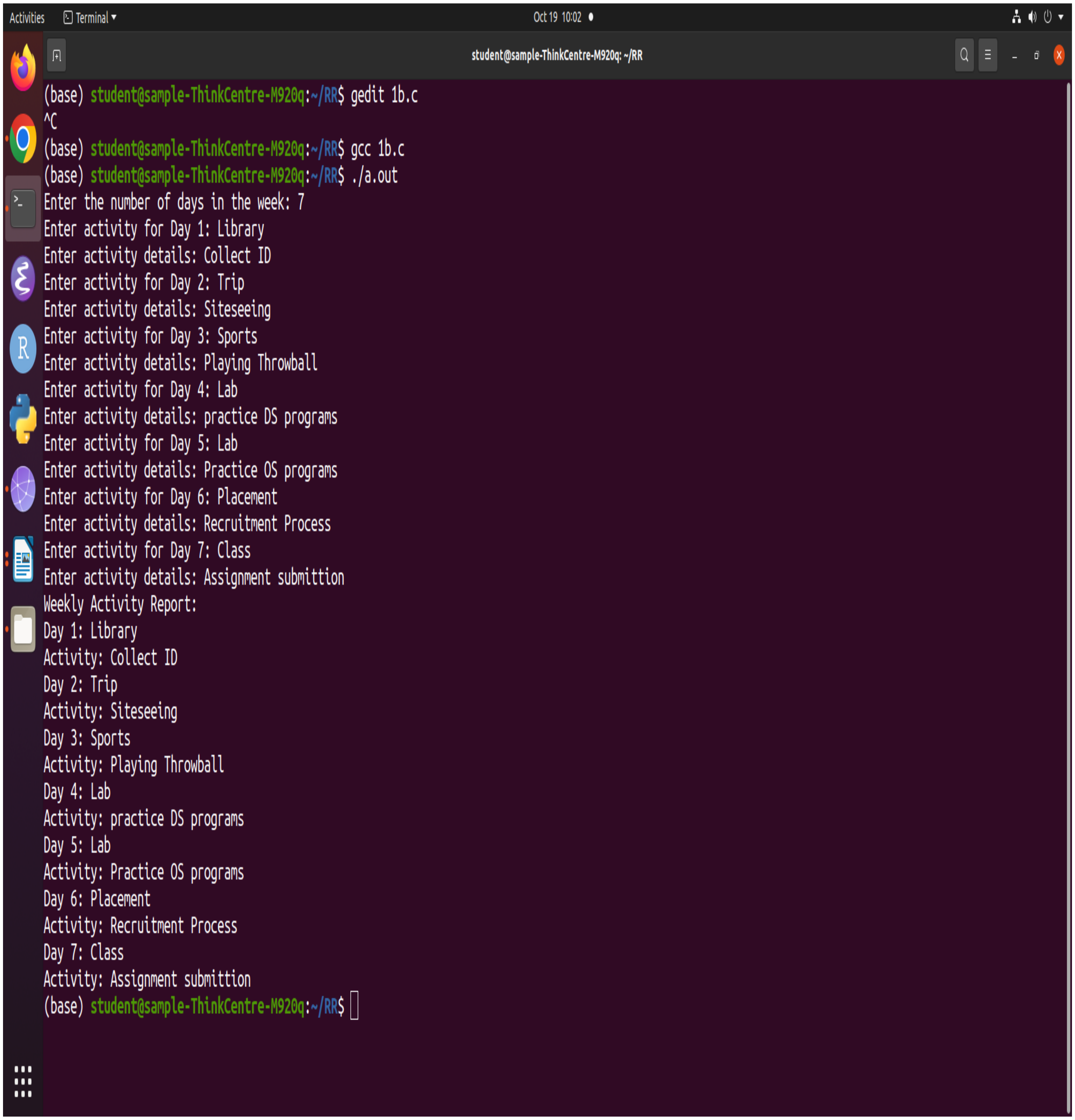
```
(base) student@sample-ThinkCentre-M920q:~$ mkdir RR
(base) student@sample-ThinkCentre-M920q:~$ cd RR
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 1.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 1.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Day: Day 1
Date: 1
Description: Activity for Day 1
Day: Day 2
Date: 2
Description: Activity for Day 2
Day: Day 3
Date: 3
Description: Activity for Day 3
Day: Day 4
Date: 4
Description: Activity for Day 4
Day: Day 5
Date: 5
Description: Activity for Day 5
Day: Day 6
Date: 6
Description: Activity for Day 6
Day: Day 7
Date: 7
Description: Activity for Day 7
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 1.c
```

```
#include <stdio.h>
#include <string.h>

struct Activity
 {
        char day[20];
        char activity[100];
};

void create(struct Activity calendar[], int numDays)
 {
        for (int i = 0; i < numDays; i++)
        {
                printf("Enter activity for Day %d: ", i + 1);
                scanf("%s", calendar[i].day);
```

```
                    getchar(); // Consume newline character
                    printf("Enter activity details: ");
                    fgets(calendar[i].activity, sizeof(calendar[i].activity), stdin);
          }
}

void read(struct Activity calendar[], int numDays)
{
          for (int i = 0; i < numDays; i++)
          {
                    printf("Enter activity for Day %d: ", i + 1);
                    scanf("%s", calendar[i].day);
                    getchar(); // Consume newline character
                    printf("Enter activity details: ");
                    fgets(calendar[i].activity, sizeof(calendar[i].activity), stdin);
          }
}

void display(struct Activity calendar[], int numDays)
{
                    printf("Weekly Activity Report:\n");
                    for (int i = 0; i < numDays; i++)
                    {
                              printf("Day %d: %s\n", i + 1, calendar[i].day);
                              printf("Activity: %s", calendar[i].activity);
                    }
}

int main()
 {
          int numDays;
          printf("Enter the number of days in the week: ");
          scanf("%d", &numDays);
           // Check for valid input
          if (numDays <= 0)
          {
                    printf("Invalid input. Please enter a positive number of days.\n");
                    return 1;
          }
           struct Activity calendar[numDays];
           create(calendar, numDays);
          display(calendar, numDays);
```

```
        return 0;
}
```

Output:

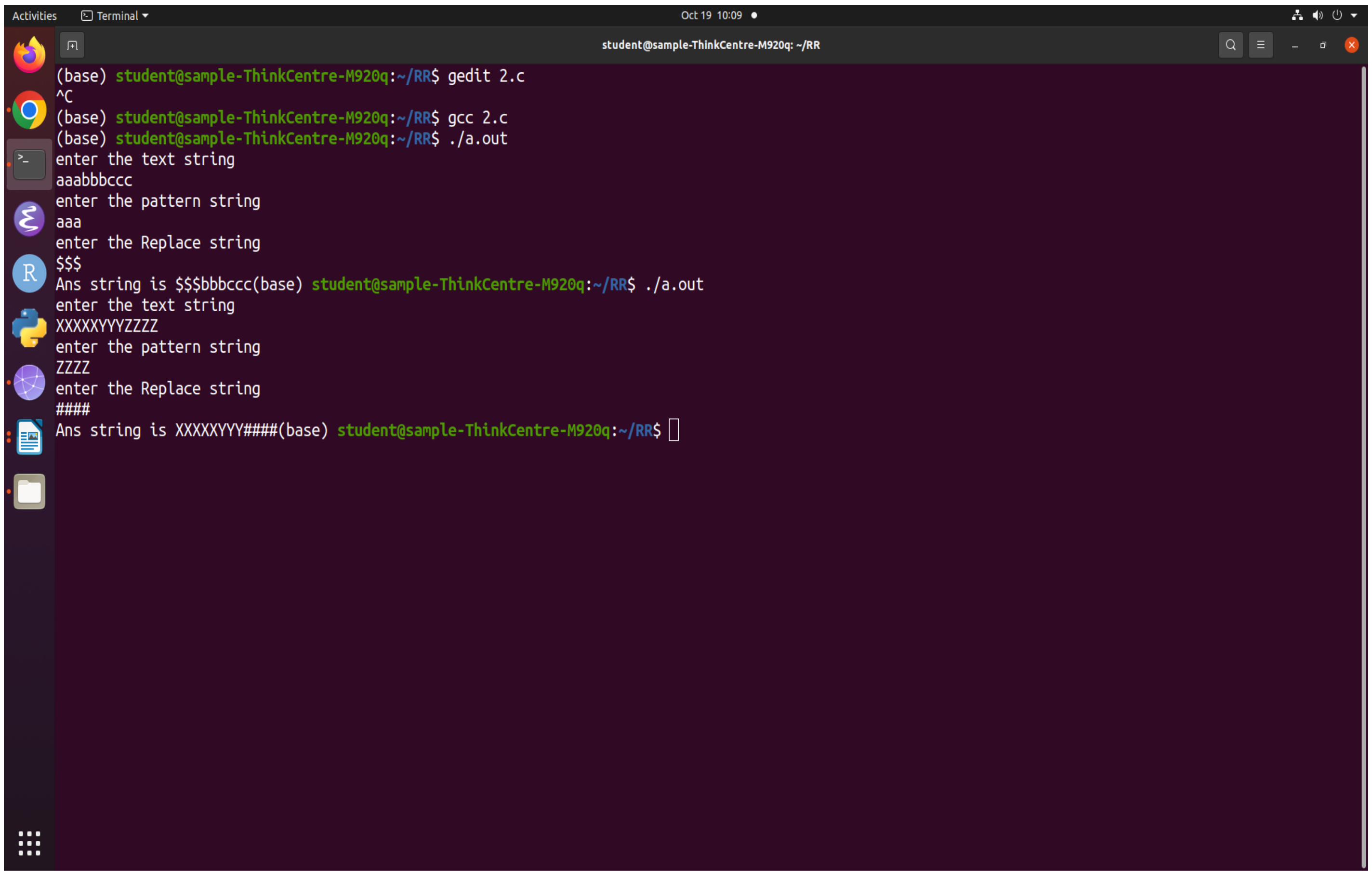


```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 1b.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 1b.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the number of days in the week: 7
Enter activity for Day 1: Library
Enter activity details: Collect ID
Enter activity for Day 2: Trip
Enter activity details: Siteseeing
Enter activity for Day 3: Sports
Enter activity details: Playing Throwball
Enter activity for Day 4: Lab
Enter activity details: practice DS programs
Enter activity for Day 5: Lab
Enter activity details: Practice OS programs
Enter activity for Day 6: Placement
Enter activity details: Recruitment Process
Enter activity for Day 7: Class
Enter activity details: Assignment submittion
Weekly Activity Report:
Day 1: Library
Activity: Collect ID
Day 2: Trip
Activity: Siteseeing
Day 3: Sports
Activity: Playing Throwball
Day 4: Lab
Activity: practice DS programs
Day 5: Lab
Activity: Practice OS programs
Day 6: Placement
Activity: Recruitment Process
Day 7: Class
Activity: Assignment submittion
(base) student@sample-ThinkCentre-M920q:~/RR$
```

2. Design, Develop and Implement a program in C for the following operations on Strings
a. Read a Main String (STR), a Pattern String (PAT) and a Replace String (REP).
b. Perform Pattern Matching Operation: Find and Replace all occurrences of PAT in STR with REP if PAT exists in STR.
Repost suitable messages in case PAT does not exist in STR.Support the program with functions for each of the above operations. Don't use built-in functions.

```
#include<stdio.h>
#include<string.h>

int i,a,j,r,flag=0;
char txt[50],pat[50],rep[50],ans[50];

void search(char  pat[ ], char  txt[ ], char rep[ ])
{
        int N = strlen(txt);
        int M = strlen(pat);
        while(i< N)
        {
                for (j = 0; j < M; j++)
                        if (txt[i+j] != pat[j])
                break;

                if (j == M)
                 {
                        r=0 ;
                        flag=1;
                        while(rep[r]!='\0')
                        {
                                ans[a++]=rep[r++] ;
                        }
                        i=i+M;
                }
                else
                        ans[a++]=txt[i++];
         }

}

void  main()
{
        printf("enter the text string \n");
        scanf("%s",txt);

        printf("enter the pattern string \n");
        scanf("%s",pat);

        printf("enter the Replace string \n");
        scanf("%s",rep);

        search(pat, txt, rep);

        if (flag==0)
                printf("pattern not found\n");
        else
                printf("Ans string is %s",ans);
}
```

Output:

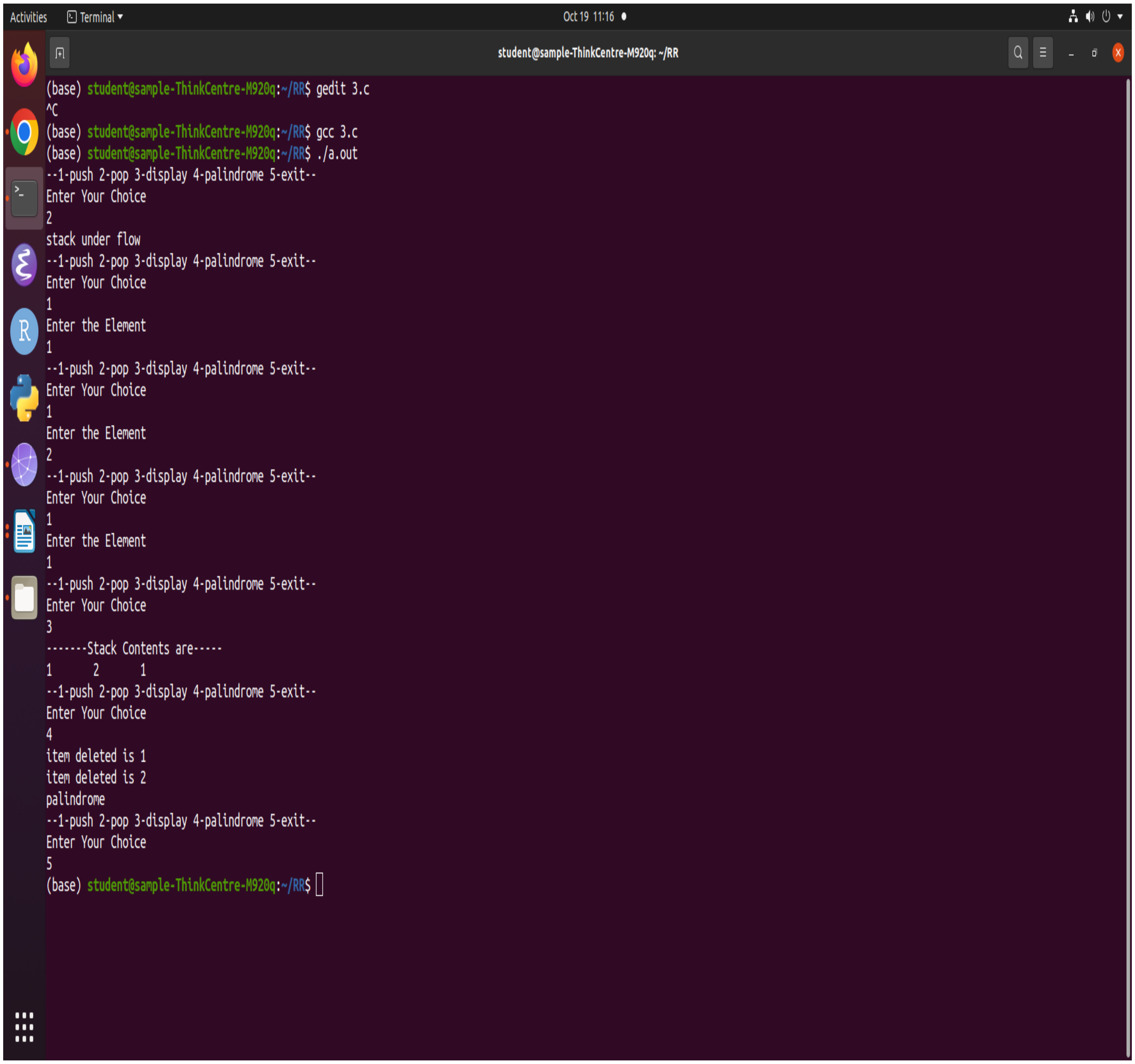


```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 2.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 2.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
enter the text string
aaabbbccc
enter the pattern string
aaa
enter the Replace string
$$$
Ans string is $$$bbbccc(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
enter the text string
XXXXXYYYZZZZ
enter the pattern string
ZZZZ
enter the Replace string
####
Ans string is XXXXXYYY####(base) student@sample-ThinkCentre-M920q:~/RR$
```

3. Design, Develop and Implement a menu driven program in C for the following operations on STACK of integers (Array implementation of stack with maximum size MAX)
a. Push an element on to stack
b. Pop an element from stack.
c. Demonstrate how stack can be used to check palindrome.
d. Demonstrate Overflow and Underflow situations on stack.
e. Display the status of stack.
f. Exit.
Support the program with appropriate functions for each of the above operations.

```
#include<stdio.h>
#include<stdlib.h>
#define SIZE 3

int top=-1,s[10],elem;

void push()
{
        printf("Enter the Element \n");
        scanf("%d",&elem);

        if(top< SIZE-1)
        {
                top++;
                s[top]=elem;
        }
        else
        printf("Stack Over Flow \n");
}

int  pop()
{
        if(top==-1)
        {
                printf("stack under flow \n");
                return 0;

        }
        else
        {
                elem=s[top];
                printf("item deleted is %d\n",elem);
                top--;
                return elem;

        }
}

void display()
{
        int i;
        if(top ==-1)
        printf("stack empty \n");

        else
        printf("-------Stack Contents are-----\n");
        for(i=0;i<=top;i++)
        {
                printf("%d\t",s[i]);

        }
        printf("\n");
}


void palindrome()
{
        int i;
        for(i=0;i<=top;i++)
        if(s[i]!=pop())
        {
                printf("Not palindrome \n");
                return;
        }
        printf("palindrome\n");
}



void main()
{
        int ch;

        for(;;)
        {
                printf("--1-push 2-pop 3-display 4-palindrome 5-exit--\n");
                printf("Enter Your Choice \n");
                scanf("%d",&ch);

                switch(ch)
                {
                        case 1:push();
                        break;

                        case 2:pop();
                        break;

                        case 3:display();
                        break;

                        case 4:palindrome();
                        break;

                        case 5:exit(0);

                }
         }
}
```

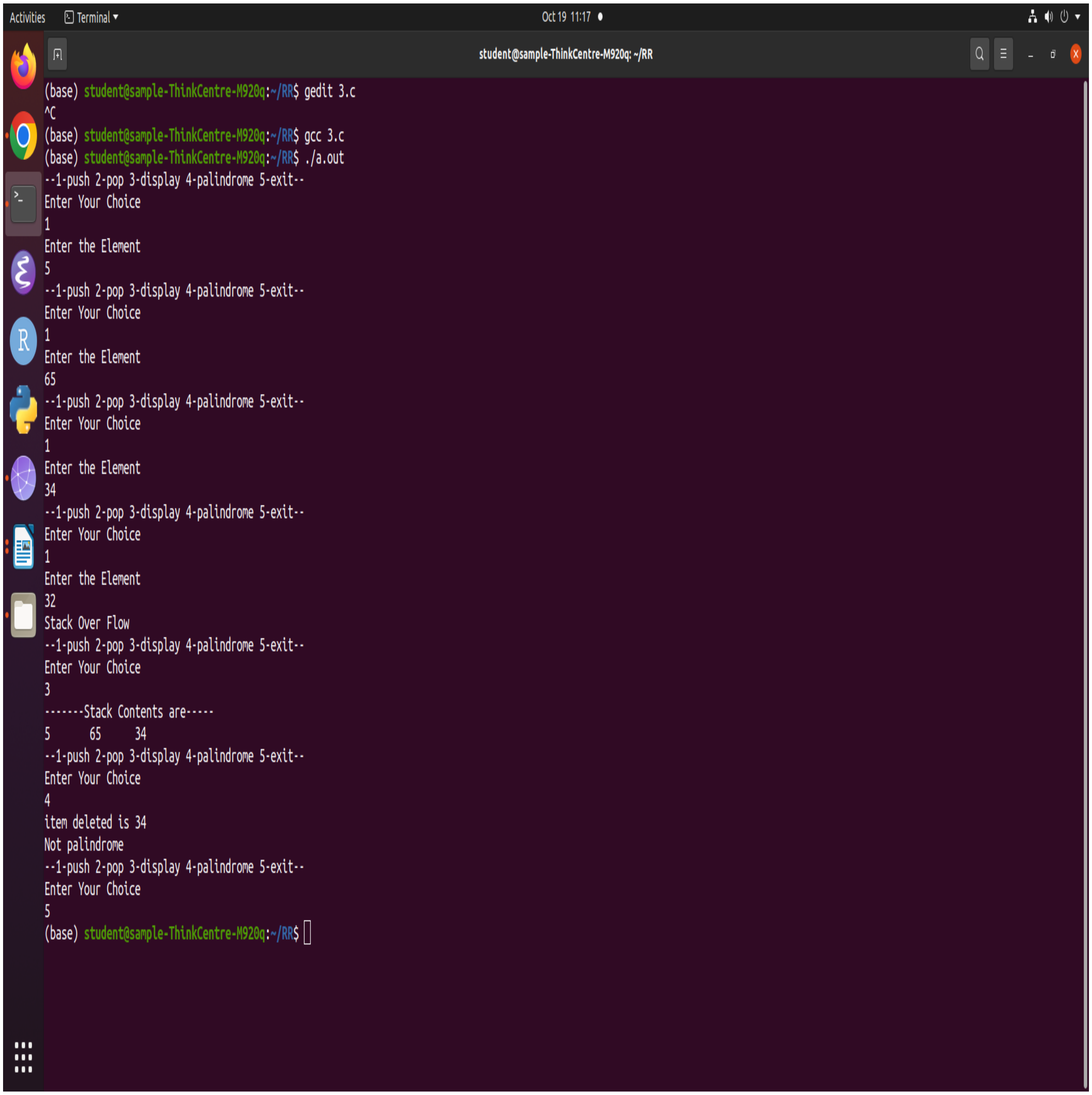
Output:



Activities Terminal ▾ Oct 19 11:16 ●

student@sample-ThinkCentre-M920q: ~/RR

```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 3.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 3.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
2
stack under flow
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
1
Enter the Element
1
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
1
Enter the Element
2
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
1
Enter the Element
1
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
3
-------Stack Contents are-----
1       2       1
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
4
item deleted is 1
item deleted is 2
palindrome
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
5
(base) student@sample-ThinkCentre-M920q:~/RR$ 
```

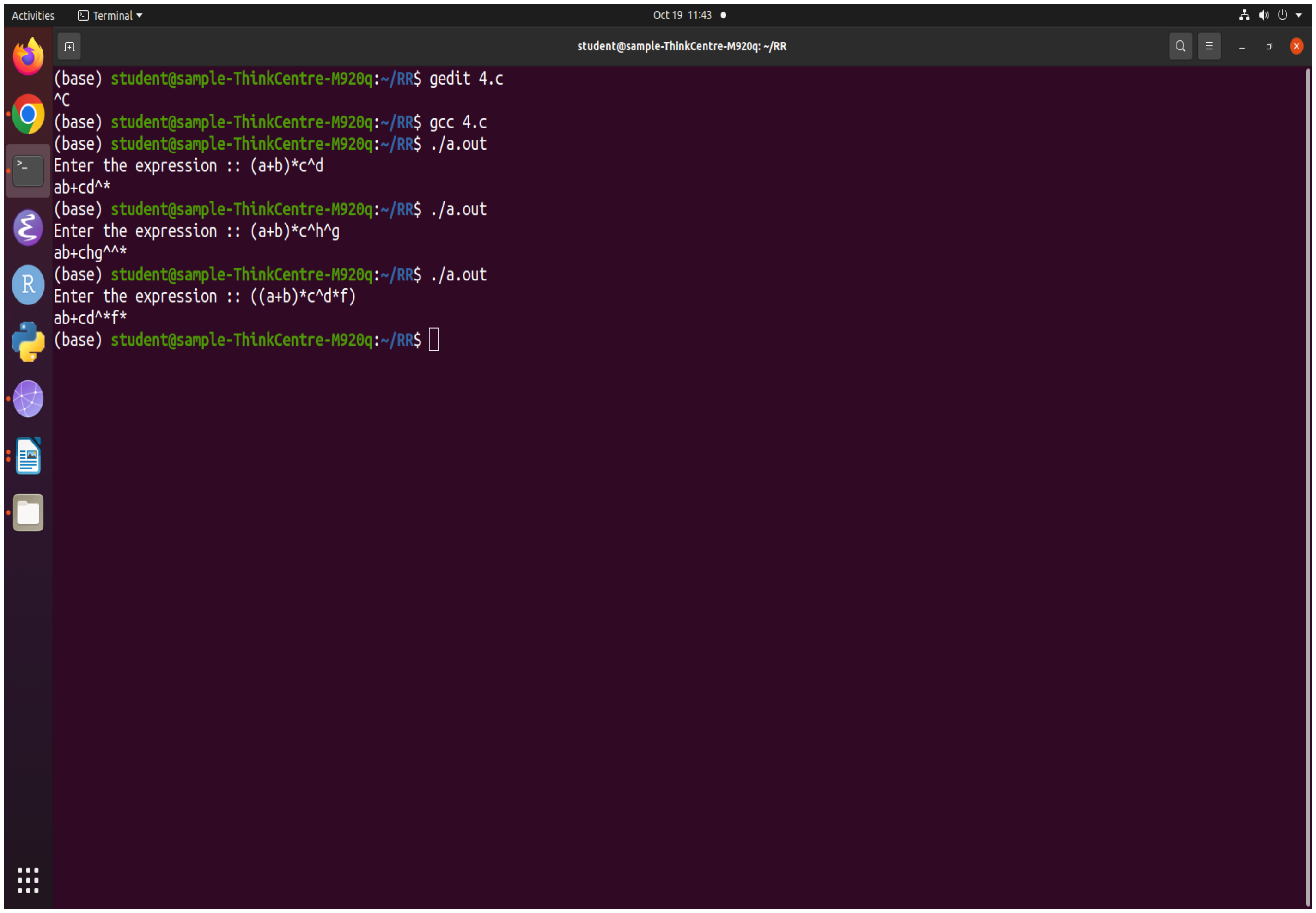


```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 3.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 3.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
1
Enter the Element
5
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
1
Enter the Element
65
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
1
Enter the Element
34
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
1
Enter the Element
32
Stack Over Flow
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
3
-------Stack Contents are-----
5       65      34
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
4
item deleted is 34
Not palindrome
--1-push 2-pop 3-display 4-palindrome 5-exit--
Enter Your Choice
5
(base) student@sample-ThinkCentre-M920q:~/RR$ 
```

4. Design, Develop and Implement a Program in C for converting an Infix Expression to Postfix Expression. Program should support for both parenthesized and free parenthesized expressions with the operators: +, -, *, /, %(Remainder), ^ (Power) and alphanumeric operands.

```
#include<stdio.h>
#include<stdlib.h>
#include<ctype.h>
char stack[20];
int top = -1;
void push(char x)
{
        stack[++top] = x;
}
char pop()
{
        if(top == -1)
                return -1;
        else
                return stack[top--];
}
int st_pri(char x)
{
        if(x == '(')
                return 0;
        if(x == '+' || x == '-'||x=='%')
                return 2;
        if(x == '*' || x == '/')
                return 4;
        if(x == '^' || x == '$')
                return 5;
return -1;
}
int exp_pri(char x)
{

        if(x == '+' || x == '-' || x=='%')
                return 1;
        if(x == '*' || x == '/')
                return 3;
        if(x == '^' || x == '$')
                return 6;
        return -1;
}
void main()
{
        char exp[20];
        char *e, x;
        printf("Enter the expression :: ");
        scanf("%s",exp);
        e = exp;
        while(*e != '\0')
        {
                if(isalnum(*e))
                        printf("%c",*e);
                else if(*e == '(')
                        push(*e);
                else if(*e == ')')
                {
                        while((x = pop()) != '(')
                                printf("%c", x);
                }
                else
                {
                        while(st_pri(stack[top]) >= exp_pri(*e))
                                printf("%c",pop());
                        push(*e);
                }
                e++;
        }
        while(top != -1)
        {
                printf("%c",pop());
        }
        printf("\n");
 }
```

Output:



```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 4.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 4.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the expression :: (a+b)*c^d
ab+cd^*
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the expression :: (a+b)*c^h^g
ab+chg^^*
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the expression :: ((a+b)*c^d*f)
ab+cd^*f*
(base) student@sample-ThinkCentre-M920q:~/RR$
```

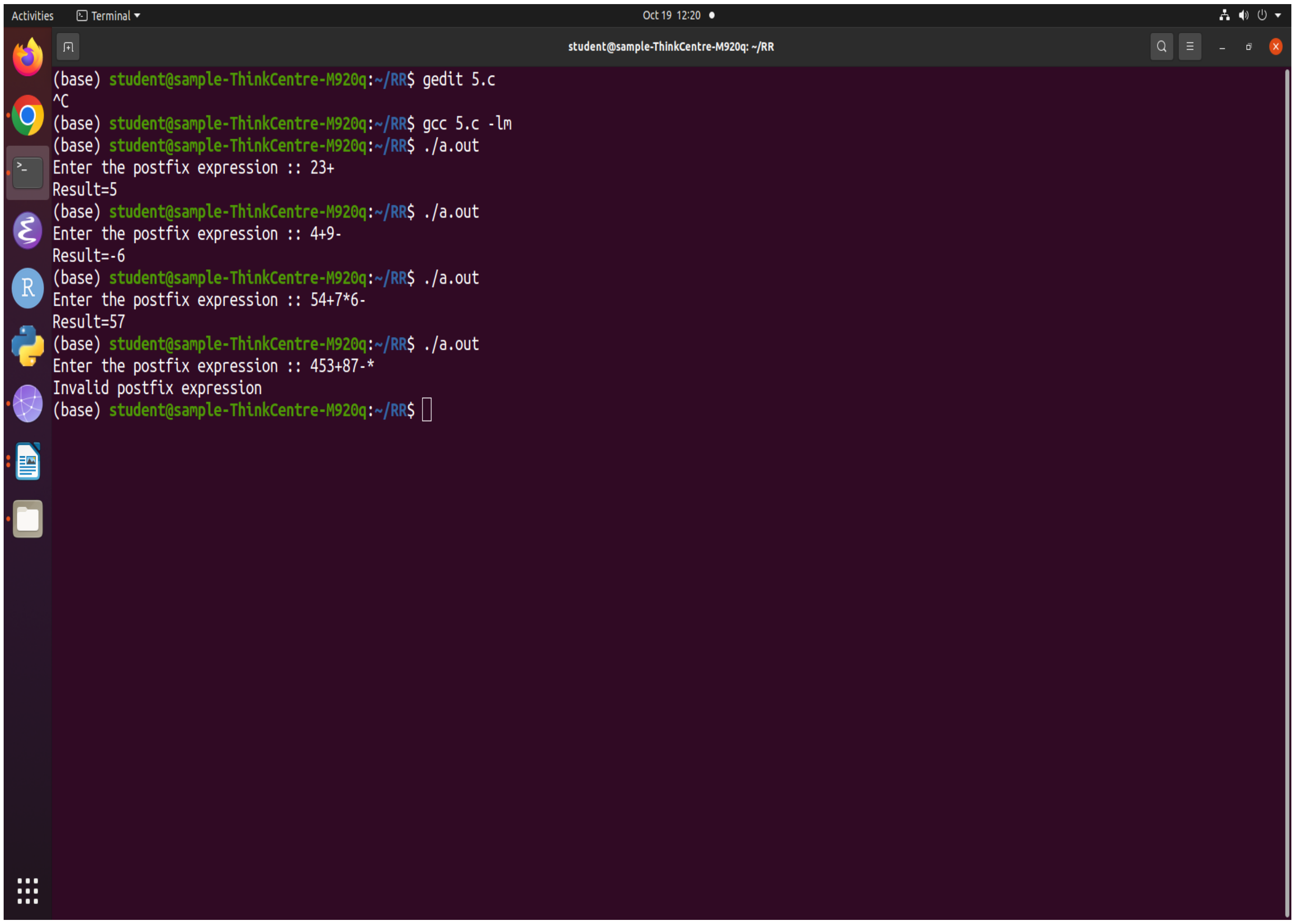
5. Design, Develop and Implement a Program in C for the following Stack Applications a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^ b.Solving Tower of Hanoi problem with n disks.

a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <ctype.h>
#include <stdlib.h>

int stack[20];
int top = -1;

void push(int x)
{
        stack[++top] = x;
}

int pop()
{
         if (top == -1)
        return -1;
        else
        return stack[top--];
}

int main()
{
        char exp[20], symb;
        int op1, op2, i;
        printf("Enter the postfix expression :: ");
        fgets(exp, sizeof(exp), stdin);

        for (i = 0; i < strlen(exp); i++)
        {
                symb = exp[i];
                if (isdigit(symb))
                push(symb - '0');
```

```
		else if (symb != ' ' && symb != '\n')
		{
			op2 = pop();
			op1 = pop();
			switch (symb)
			{
				case '+':
					push(op1 + op2);
					break;
				case '-':
					push(op1 - op2);
					break;
				case '*':
					push(op1 * op2);
					break;
				case '/':
					if (op2 != 0) // Check for division by zero
					push(op1 / op2);
					else
				 {
					printf("Division by zero\n");
					exit(1); // Terminate the program
				}
				break;
				case '^':
				push((int)pow(op1, op2)); // Cast the result of 'pow' to an integer
				break;
				default:
				printf("Invalid character in the expression: %c\n", symb);
				exit(1); // Terminate the program
			}
		}
	}

	if (top == 0) // Check for a valid postfix expression
	printf("Result=%d\n", stack[top]);
	else
	printf("Invalid postfix expression\n");

	return 0;
}
```

Output:



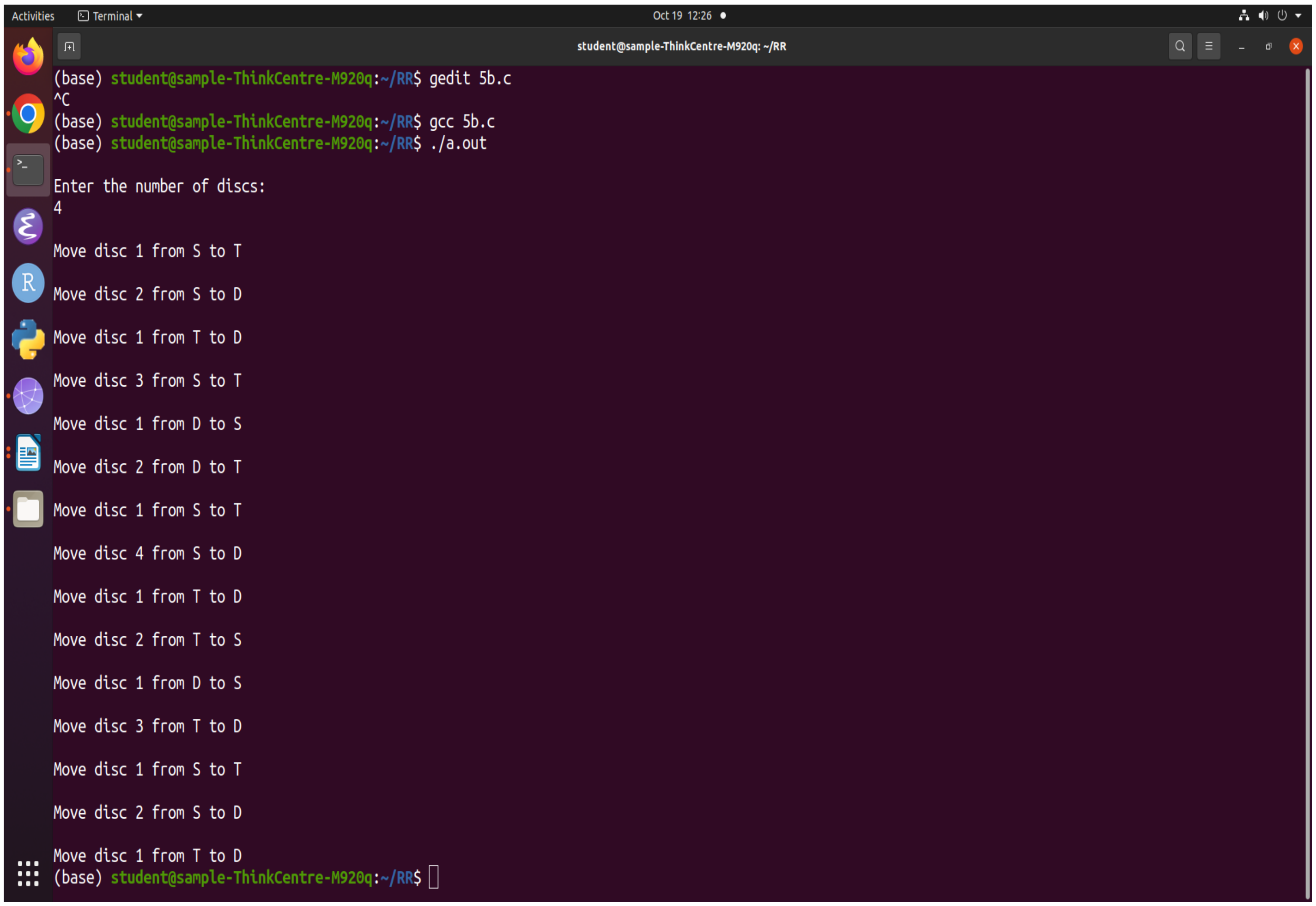
```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 5.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 5.c -lm
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the postfix expression :: 23+
Result=5
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the postfix expression :: 4+9-
Result=-6
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the postfix expression :: 54+7*6-
Result=57
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the postfix expression :: 453+87-*
Invalid postfix expression
(base) student@sample-ThinkCentre-M920q:~/RR$
```

b. Solving Tower *of Hanoi* problem with n disks.

```
#include <stdio.h>

void tower(int n, char S, char T, char D)
{
   if (n == 0)
      return;
   tower(n - 1, S, D, T);
```

```
    printf("\nMove disc %d from %c to %c\n", n, S, D);
    tower(n - 1, T, S, D);
}
int main()
{
    int n;
    printf("\nEnter the number of discs: \n");
    scanf("%d", &n);
    tower(n, 'S', 'T', 'D');
    return 0;
}
```

Output:



```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 5b.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 5b.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out

Enter the number of discs:
4

Move disc 1 from S to T

Move disc 2 from S to D

Move disc 1 from T to D

Move disc 3 from S to T

Move disc 1 from D to S

Move disc 2 from D to T

Move disc 1 from S to T

Move disc 4 from S to D

Move disc 1 from T to D

Move disc 2 from T to S

Move disc 1 from D to S

Move disc 3 from T to D

Move disc 1 from S to T

Move disc 2 from S to D

Move disc 1 from T to D
(base) student@sample-ThinkCentre-M920q:~/RR$
```

6. Design, Develop and Implement a menu driven Program in C for the following

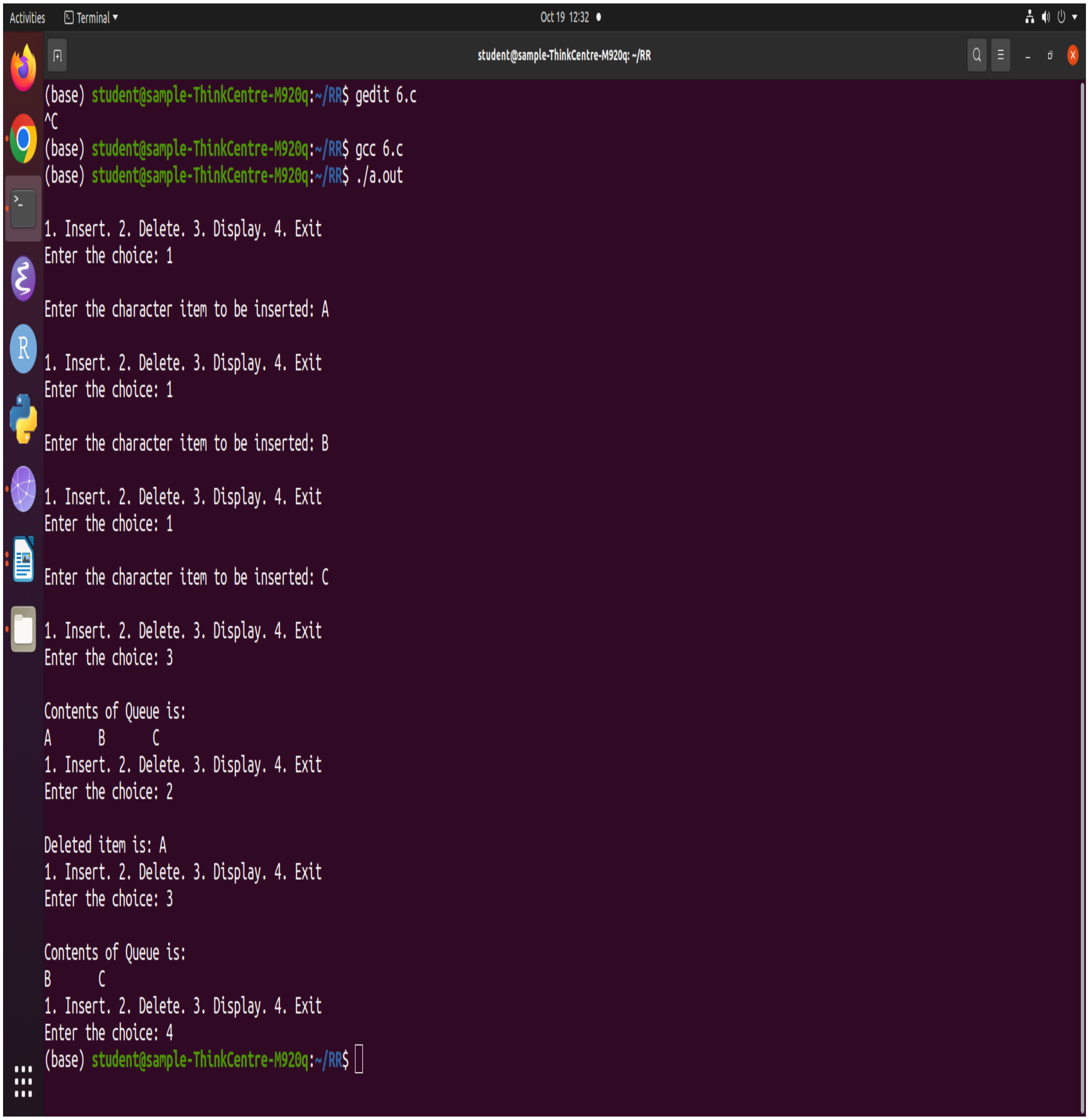
operations on Circular
QUEUE of Characters (Array Implementation of Queue with maximum size MAX)
a. Insert an Element on to Circular QUEUE
b. Delete an Element from Circular QUEUE
c. Demonstrate *Overflow* and *Underflow* situations on Circular QUEUE
d. Display the status of Circular QUEUE
e. Exit
Support the program with appropriate functions for each of the above operations

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 4

int front = 0, rear = -1, count = 0;
char q[MAX], item;

void insert()
{
        printf("\nEnter the character item to be inserted: ");
        while (getchar() != '\n');
        scanf("%c", &item);
        if (count == MAX)
        {
                printf("\nQueue is Full");
                return;
        }
        rear = (rear + 1) % MAX;
        q[rear] = item;
        count++;
}

void del()
{
        if (count == 0)
        {
                printf("\nQueue is Empty");
                return;
        }
        item = q[front];
        printf("\nDeleted item is: %c", item);
        front = (front + 1) % MAX;
        count--;
```

```
}

void display()
{
         int i, f = front;
        if (count == 0)
        {
                printf("\nQueue is Empty");
                return;
        }
        printf("\nContents of Queue is:\n");
        for (i = 0; i < count; i++)
        {
                printf("%c\t", q[f]);
                f = (f + 1) % MAX;
        }
}

int main()
{
        int ch;
        for (;;)
        {
                printf("\n1. Insert. 2. Delete. 3. Display. 4. Exit ");
                printf("\nEnter the choice: ");
                scanf("%d", &ch);
                switch (ch)
                {
                        case 1:
                        insert();
                        break;
                        case 2:
                        del();
                        break;
                        case 3:
                        display();
                        break;
                        case 4:
                        exit(0);
                        break;
                        default:
                        printf("\nInvalid choice. Please try again.\n");
                }
```

```
	}
	return 0;
}
```

Output:



```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 6.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 6.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out

1. Insert. 2. Delete. 3. Display. 4. Exit
Enter the choice: 1

Enter the character item to be inserted: A

1. Insert. 2. Delete. 3. Display. 4. Exit
Enter the choice: 1

Enter the character item to be inserted: B

1. Insert. 2. Delete. 3. Display. 4. Exit
Enter the choice: 1

Enter the character item to be inserted: C

1. Insert. 2. Delete. 3. Display. 4. Exit
Enter the choice: 3

Contents of Queue is:
A       B       C
1. Insert. 2. Delete. 3. Display. 4. Exit
Enter the choice: 2

Deleted item is: A
1. Insert. 2. Delete. 3. Display. 4. Exit
Enter the choice: 3

Contents of Queue is:
B       C
1. Insert. 2. Delete. 3. Display. 4. Exit
Enter the choice: 4
(base) student@sample-ThinkCentre-M920q:~/RR$
```

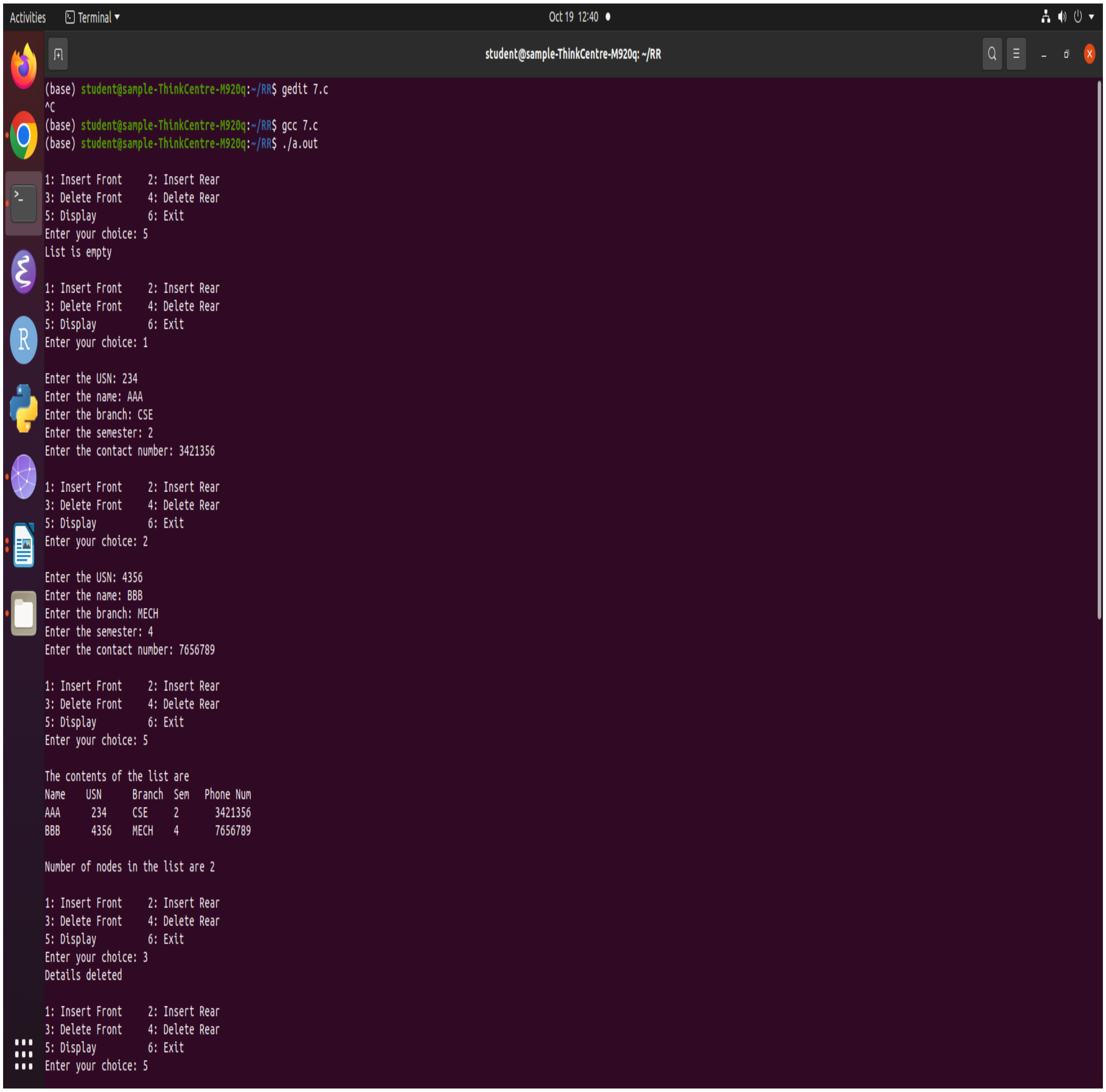
7. Design, Develop and Implement a menu driven Program in C for the following operations on Singly Linked List (SLL) of Student Data with the fields: *USN, Name, Branch, Sem, PhNo*
a. Create a SLL of N Students Data by using *front insertion*.
b. Display the status of SLL and count the number of nodes in it
c. Perform Insertion and Deletion at End of SLL
d. Perform Insertion and Deletion at Front of SLL
e. Demonstrate how this SLL can be used as STACK and QUEUE
f. Exit

```
#include <stdio.h>
#include <stdlib.h>

struct node
{
        int usn;
        char name[25];
        char branch[25];
        int sem;
        long int phno;
        struct node* link;
};

typedef struct node* NODE;

NODE getnode()
{
        NODE temp;
        temp = (NODE)malloc(sizeof(struct node));
        if (temp == NULL)
        {
                printf("Insufficient Memory\n");
                exit(0);
        }
        printf("\nEnter the USN: ");
        scanf("%d", &temp->usn);
        printf("Enter the name: ");
        scanf("%s", temp->name);
        printf("Enter the branch: ");
        scanf("%s", temp->branch);
        printf("Enter the semester: ");
        scanf("%d", &temp->sem);
        printf("Enter the contact number: ");
        scanf("%ld", &temp->phno);
        temp->link = NULL;
        return temp;
}

NODE insert_front(NODE first)
{
        NODE temp;
        temp = getnode();
        temp->link = first;
        return temp;
}

NODE insert_rear(NODE first)
{
        NODE temp, cur;
        temp = getnode();

        if (first == NULL)
           return temp;

        cur = first;
        while (cur->link != NULL)
        cur = cur->link;
        cur->link = temp;
        return first;
}

void display(NODE first)
{
        NODE cur;
        int count = 0;
        if (first == NULL)
        {
                printf("List is empty\n");
                return;
        }
        printf("\nThe contents of the list are\n");
        printf("Name    USN     Branch  Sem   Phone Num\n");
        cur = first;
        while (cur != NULL)
        {
                printf("%s\t %d\t %s\t %d\t %ld\n", cur->name, cur->usn, cur->branch, cur->sem, cur->phno);
                cur = cur->link;
                count++;
        }
        printf("\nNumber of nodes in the list are %d\n", count);
}

NODE delete_front(NODE first)
{
        NODE temp;
        if (first == NULL)
        {
                printf("\nList is empty\n");
                return first;
        }
        temp = first;
        first = first->link;
        printf("Details deleted\n");
        free(temp);
        return first;
}

NODE delete_rear(NODE first)
{
        NODE cur, prev;
        if (first == NULL)
        {
                printf("\nList is empty\n");
                return first;
        }
        if (first->link == NULL)
        {
                printf("Details deleted\n");
                free(first);
                return NULL;
        }
        prev = NULL;
        cur = first;
        while (cur->link != NULL)
        {
                prev = cur;
                cur = cur->link;
        }
```

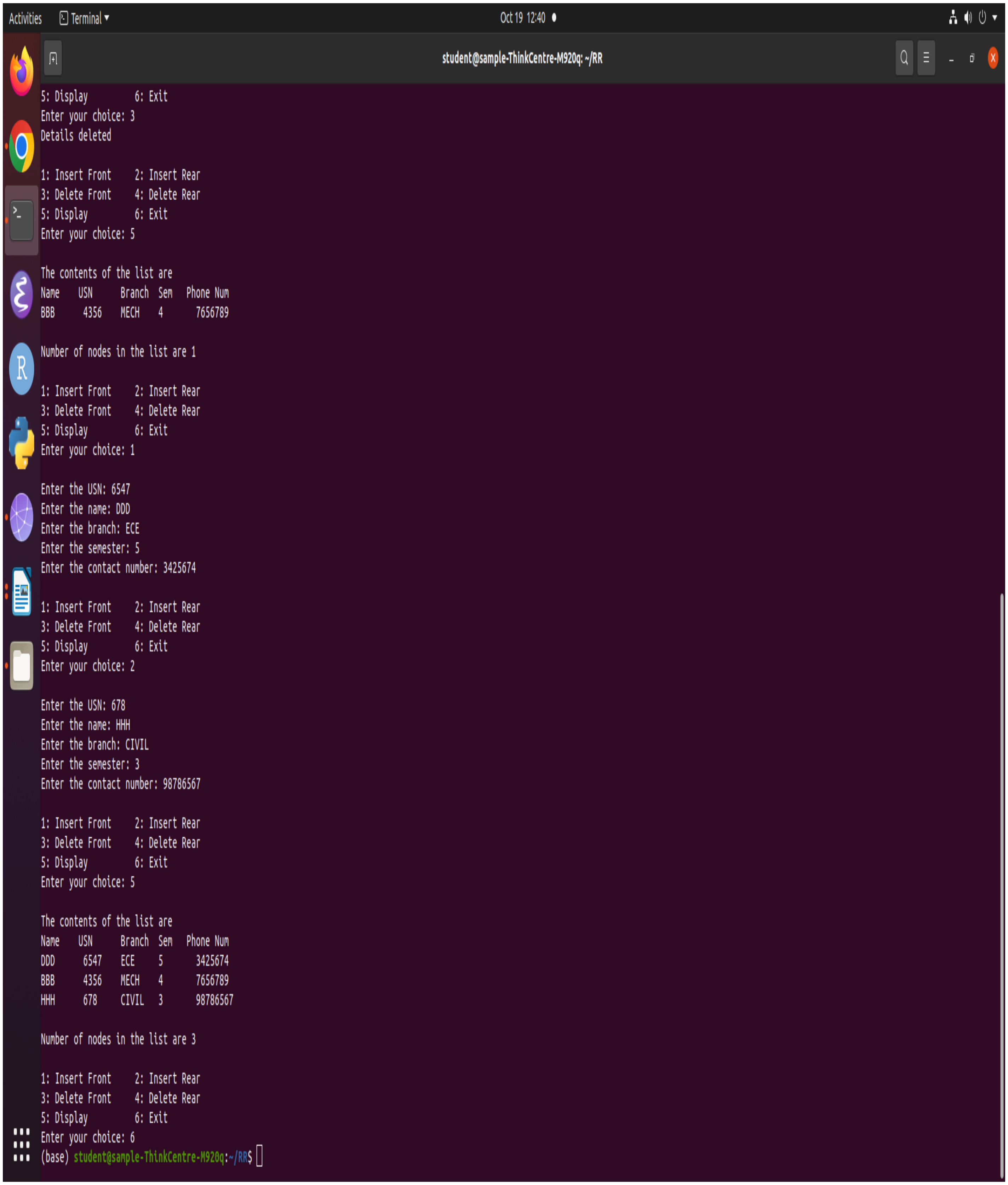
```
	printf("Details deleted\n");
	free(cur);
	prev->link = NULL;
	return first;
}

int main()
{
	NODE first = NULL;
	int ch;
	for (;;)
	{
		printf("\n1: Insert Front     2: Insert Rear\n");
		printf("3: Delete Front     4: Delete Rear\n");
		printf("5: Display          6: Exit\n");
		printf("Enter your choice: ");
		scanf("%d", &ch);
		switch (ch)
		{
			case 1:
			first = insert_front(first);
			break;

			case 2:
			first = insert_rear(first);
			break;

			case 3:
			first = delete_front(first);
			break;

			case 4:
			first = delete_rear(first);
			break;

			case 5:
			display(first);
			break;

			case 6:
			exit(0);
		}
	}
```

```
        return 0;
}
```

Output:



```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 7.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 7.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 5
List is empty

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 1

Enter the USN: 234
Enter the name: AAA
Enter the branch: CSE
Enter the semester: 2
Enter the contact number: 3421356

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 2

Enter the USN: 4356
Enter the name: BBB
Enter the branch: MECH
Enter the semester: 4
Enter the contact number: 7656789

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 5

The contents of the list are
Name    USN     Branch  Sem   Phone Num
AAA      234     CSE     2      3421356
BBB      4356    MECH    4      7656789

Number of nodes in the list are 2

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 3
Details deleted

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 5
```



```
5: Display        6: Exit
Enter your choice: 3
Details deleted

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 5

The contents of the list are
Name    USN     Branch  Sem   Phone Num
BBB     4356    MECH    4      7656789

Number of nodes in the list are 1

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 1

Enter the USN: 6547
Enter the name: DDD
Enter the branch: ECE
Enter the semester: 5
Enter the contact number: 3425674

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 2

Enter the USN: 678
Enter the name: HHH
Enter the branch: CIVIL
Enter the semester: 3
Enter the contact number: 98786567

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 5

The contents of the list are
Name    USN     Branch  Sem   Phone Num
DDD     6547    ECE     5      3425674
BBB     4356    MECH    4      7656789
HHH     678     CIVIL   3      98786567

Number of nodes in the list are 3

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
Enter your choice: 6
(base) student@sample-ThinkCentre-M920q:~/RR$
```

8.Design, Develop and Implement a menu driven Program in C for the following operations on Doubly Linked List (DLL) of Employee Data with the fields: *SSN, Name, Dept, Designation, Sal, PhNo*

a. Create a DLL of N Employees Data by using *end insertion*.
b. Display the status of DLL and count the number of nodes in it
c. Perform Insertion and Deletion at End of DLL
d. Perform Insertion and Deletion at Front of DLL
e. Demonstrate how this DLL can be used as Double Ended Queue
f. Exit

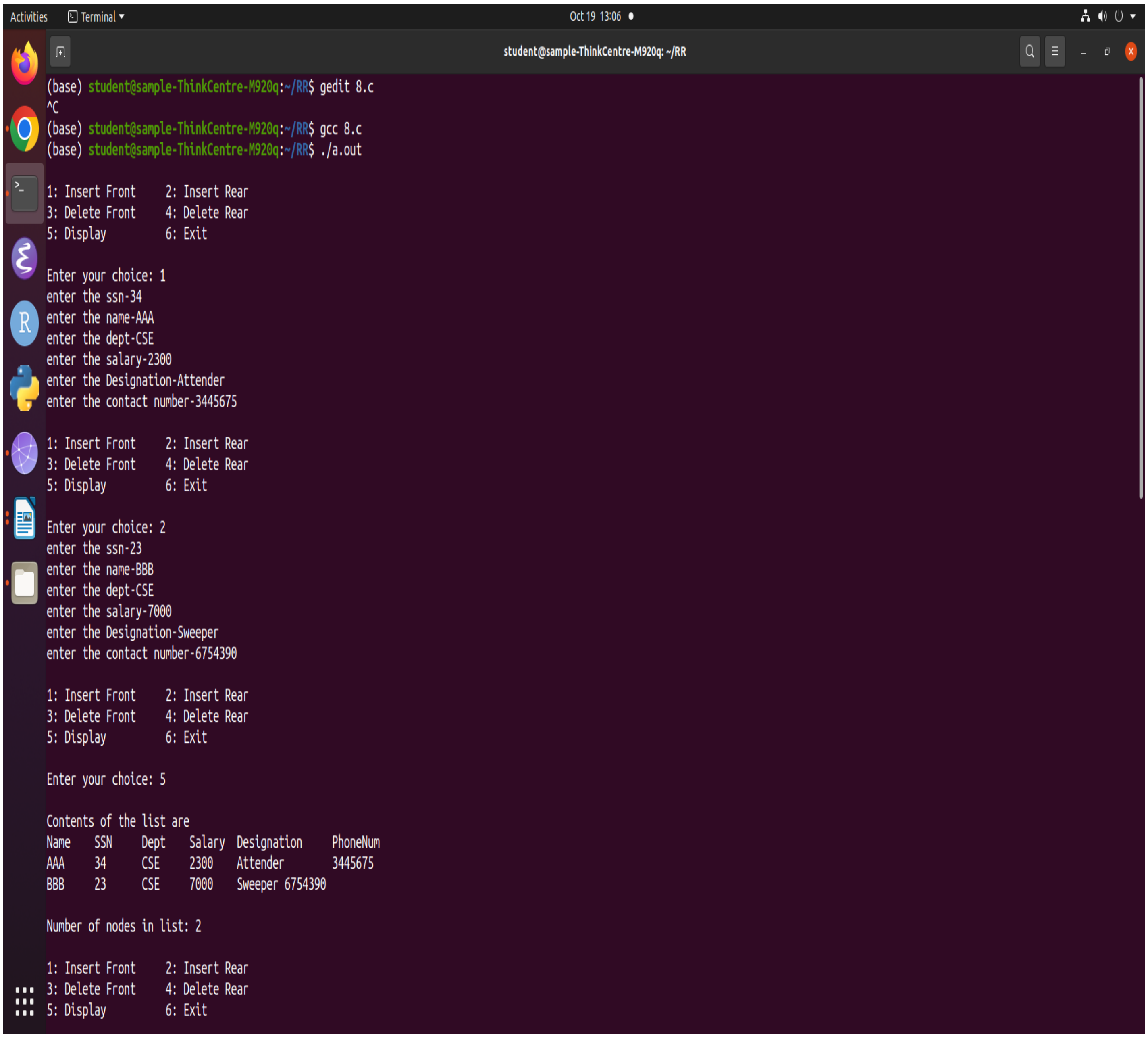
```
#include <stdio.h>
#include <stdlib.h>

struct node
{
        int ssn;
        char name[25];
        char dept[25];
        int salary;
        char desig[25];
        long int phno;
        struct node *llink;
        struct node *rlink;
};

typedef struct node *NODE;

NODE getnode()
{
        NODE temp;
         temp = (NODE)malloc(sizeof(struct node));

        if (temp == NULL)
        {
                printf("insufficient Memory\n");
                exit(0);
        }

        printf("enter the ssn-");
        scanf("%d", &temp->ssn);
        printf("enter the name-");
        scanf("%s", temp->name);
```

```
	printf("enter the dept-");
	scanf("%s", temp->dept);
	printf("enter the salary-");
	scanf("%d", &temp->salary);
	printf("enter the Designation-");
	scanf("%s", temp->desig);
	printf("enter the contact number-");
	scanf("%ld", &temp->phno);
	temp->llink = NULL;
	temp->rlink = NULL;

	return temp;
}

NODE insert_front(NODE first)
{
	NODE temp;
	temp = getnode();
	if (first == NULL)
	{
		return temp;
	}
	temp->rlink = first;
	first->llink = temp;
	return temp;
}

NODE insert_rear(NODE first)
{
	NODE temp, cur;
	temp = getnode();

	if (first == NULL)
	{
		  return temp;
	}

	cur = first;
	while (cur->rlink != NULL)
	{
		cur = cur->rlink;
	}
```

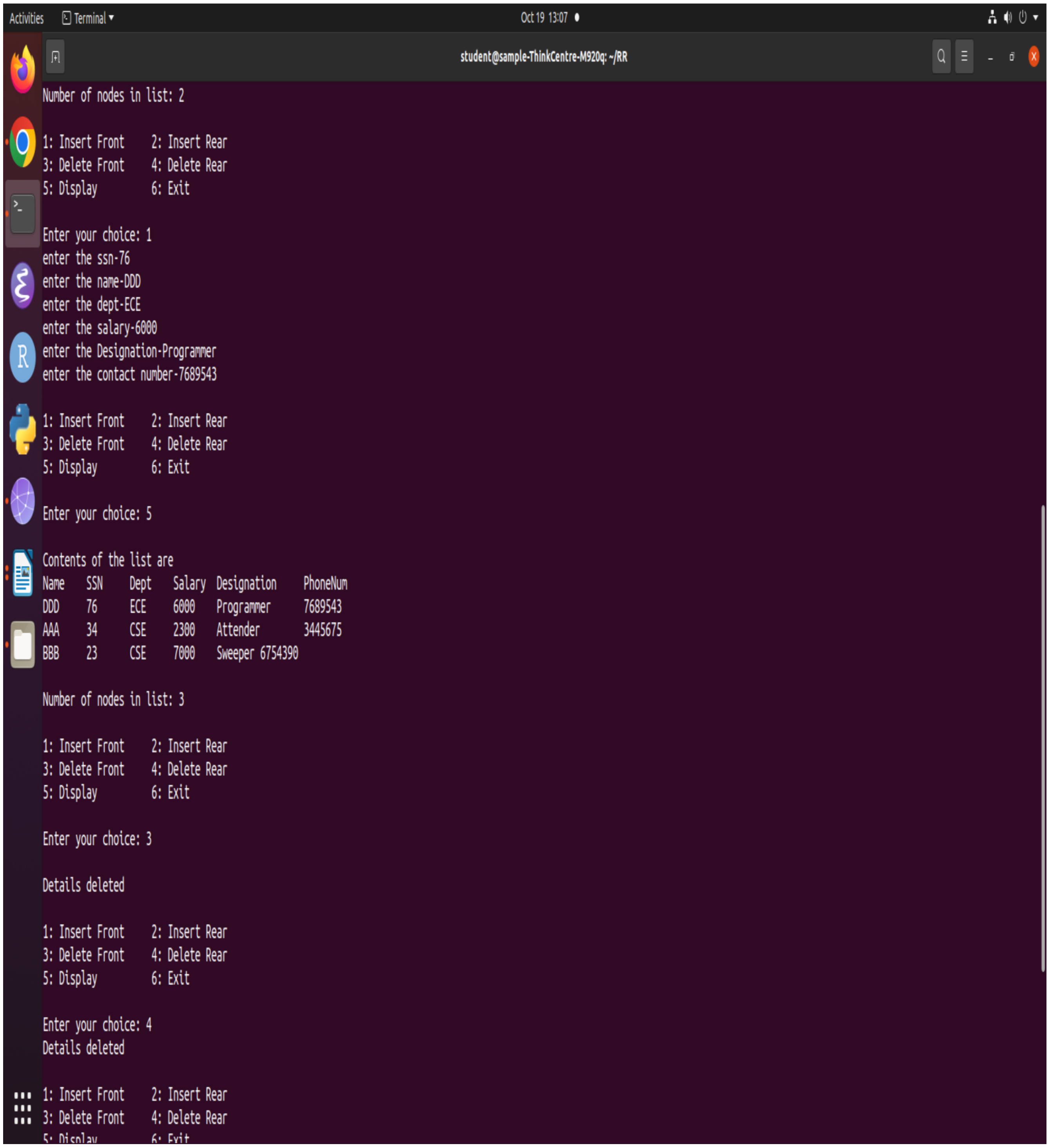
```
        cur->rlink = temp;
        temp->llink = cur;
         return first;
}

void display(NODE first)
{
        NODE cur;
        int count = 0;
        if (first == NULL)
        {
                printf("List is empty\n");
                return;
        }
        printf("\nContents of the list are\n");
        printf("Name\tSSN\tDept\tSalary\tDesignation\tPhoneNum\n");
        cur = first;
        while (cur != NULL)
        {
                printf("%s\t%d\t%s\t%d\t%s\t%ld\n", cur->name, cur->ssn, cur->dept, cur->salary, cur-
                                                >desig, cur->phno);
                cur = cur->rlink;
                count++;
        }
        printf("\nNumber of nodes in list: %d\n", count);
}

NODE delete_front(NODE first)
{
        NODE temp;
        if (first == NULL)
         {
                printf("\nList is empty\n");
                return first;
        }

        if (first->rlink == NULL)
        {
                printf("Details deleted\n");
                free(first);
                return NULL;
        }
        temp = first->rlink;
        temp->llink = NULL;
        printf("\nDetails deleted\n");
        free(first);
        return temp;
}

NODE delete_rear(NODE first)
{
        NODE cur, prev;

        if (first == NULL)
        {
                printf("\nList is empty\n");
                return first;
        }

        if (first->rlink == NULL)
        {
                printf("Details deleted\n");
                free(first);
                return NULL;
        }

        prev = NULL;
        cur = first;

        while (cur->rlink != NULL)
        {
                prev = cur;
                cur = cur->rlink;
        }

        printf("Details deleted\n");
        free(cur);

        prev->rlink = NULL;
        return first;
}

int main()
{
        NODE first = NULL;
        int ch, item, i;
```

```
	for (;;)
	{
		printf("\n1: Insert Front     2: Insert Rear\n");
		printf("3: Delete Front     4: Delete Rear\n");
		printf("5: Display          6: Exit\n");

		printf("\nEnter your choice: ");
		scanf("%d", &ch);

		switch (ch)
		{
			case 1:
				first = insert_front(first);
			break;
			case 2:
				first = insert_rear(first);
			break;
			case 3:
				first = delete_front(first);
			break;
			case 4:
				first = delete_rear(first);
			break;
			case 5:
			display(first);
			break;
			case 6:
			exit(0);
		}
	}

	return 0;
}
```

Output:



```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 8.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 8.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit

Enter your choice: 1
enter the ssn-34
enter the name-AAA
enter the dept-CSE
enter the salary-2300
enter the Designation-Attender
enter the contact number-3445675

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit

Enter your choice: 2
enter the ssn-23
enter the name-BBB
enter the dept-CSE
enter the salary-7000
enter the Designation-Sweeper
enter the contact number-6754390

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit

Enter your choice: 5

Contents of the list are
Name    SSN     Dept    Salary  Designation     PhoneNum
AAA     34      CSE     2300    Attender        3445675
BBB     23      CSE     7000    Sweeper 6754390

Number of nodes in list: 2

1: Insert Front     2: Insert Rear
3: Delete Front     4: Delete Rear
5: Display          6: Exit
```



```
Number of nodes in list: 2

1: Insert Front    2: Insert Rear
3: Delete Front    4: Delete Rear
5: Display         6: Exit

Enter your choice: 1
enter the ssn-76
enter the name-DDD
enter the dept-ECE
enter the salary-6000
enter the Designation-Programmer
enter the contact number-7689543

1: Insert Front    2: Insert Rear
3: Delete Front    4: Delete Rear
5: Display         6: Exit

Enter your choice: 5

Contents of the list are
Name    SSN     Dept    Salary  Designation     PhoneNum
DDD     76      ECE     6000    Programmer      7689543
AAA     34      CSE     2300    Attender        3445675
BBB     23      CSE     7000    Sweeper 6754390

Number of nodes in list: 3

1: Insert Front    2: Insert Rear
3: Delete Front    4: Delete Rear
5: Display         6: Exit

Enter your choice: 3

Details deleted

1: Insert Front    2: Insert Rear
3: Delete Front    4: Delete Rear
5: Display         6: Exit

Enter your choice: 4
Details deleted

1: Insert Front    2: Insert Rear
3: Delete Front    4: Delete Rear
5: Display         6: Exit
```

9. Design, Develop and Implement a Program in C for the following operations on Singly Circular Linked List (SCLL) with header nodes
a. Represent and Evaluate a Polynomial P(x,y,z) = 6x2y2z-4yz5+3x3yz+2xy5z-2xyz3
b. Find the sum of two polynomials POLY1(x,y,z) and POLY2(x,y,z) and store the result in POLYSUM(x,y,z) Support the program with appropriate functions for each of the above operations.

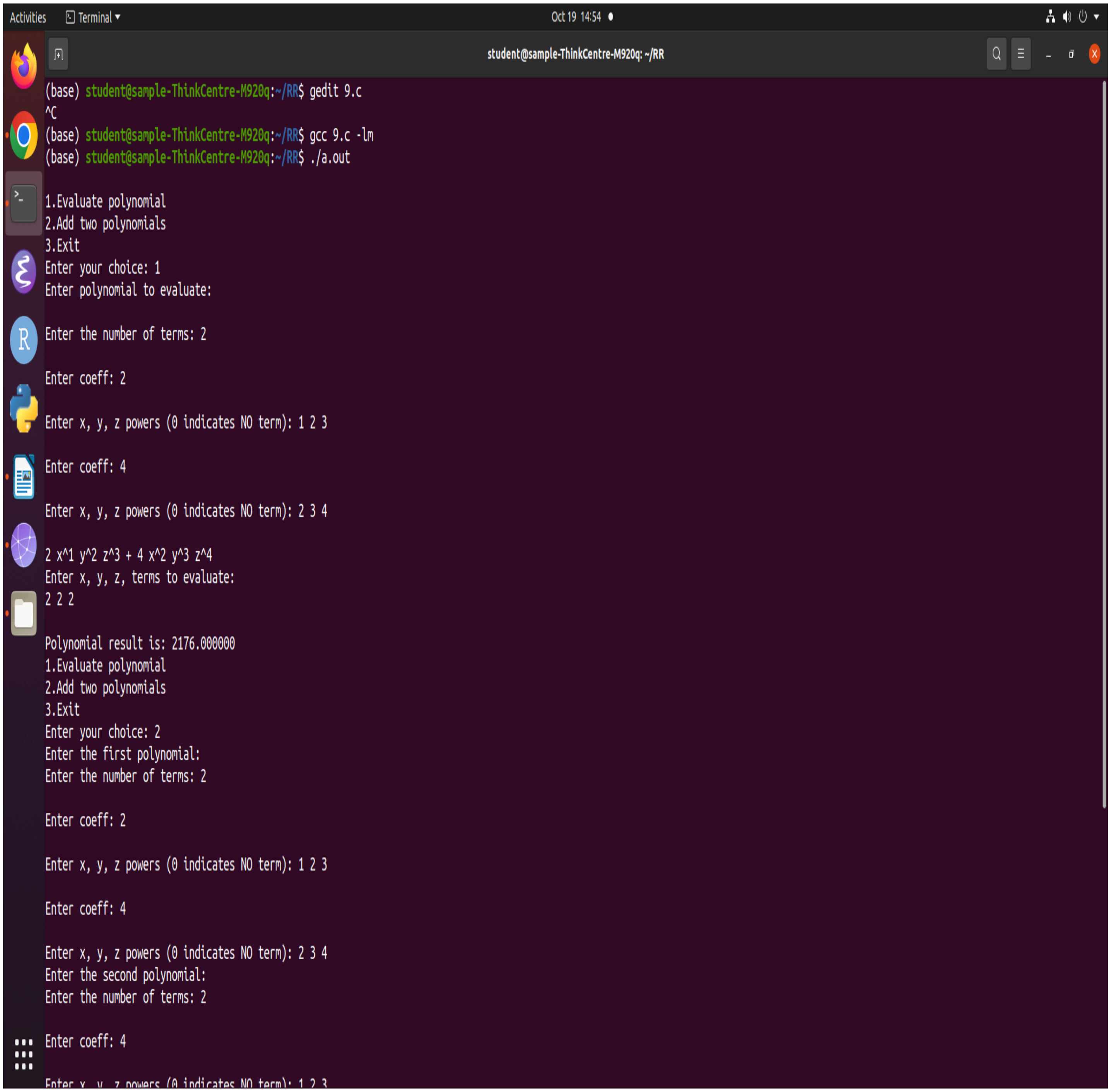
```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

struct node
{
	int cf, px, py, pz;
	int flag;
	struct node* link;
};

typedef struct node NODE;

NODE* getnode()
{
	NODE* x;
	x = (NODE*)malloc(sizeof(NODE));
	if (x == NULL)
	{
		printf("Insufficient memory\n");
		exit(0);
	}
	return x;
}

void display(NODE* start)
{
	NODE* temp;
	if (start->link == start)
	{
		printf("Polynomial does not exist\n");
		return;
	}
	temp = start->link;
	printf("\n");
	while (temp != start)
	{
		printf("%d x^%d y^%d z^%d", temp->cf, temp->px, temp->py, temp->pz);
		if (temp->link != start)
		printf(" + ");
		temp = temp->link;
	}
}

NODE* insert_rear(int cf, int x, int y, int z, NODE* start)
{
	NODE* temp, *cur;
	temp = getnode();

	temp->cf = cf;
	temp->px = x;
	temp->py = y;
	 temp->pz = z;

	cur = start->link;

	while (cur->link != start)
	{
		cur = cur->link;
	}
	cur->link = temp;
	temp->link = start;

	return start;
}

NODE* read_poly(NODE* start)
{
	int px, py, pz, cf, ch;
	int n, i;

	printf("\nEnter the number of terms: ");
	scanf("%d", &n);

	for (i = 0; i < n; i++)
	{
		printf("\nEnter coeff: ");
		scanf("%d", &cf);
		printf("\nEnter x, y, z powers (0 indicates NO term): ");
		scanf("%d%d%d", &px, &py, &pz);

		start = insert_rear(cf, px, py, pz, start);
	}
	return start;
}

void evaluate(NODE* start)
{
	NODE* h1;
	int x, y, z;
	float result = 0.0;

	h1 = start->link;

	printf("\nEnter x, y, z, terms to evaluate:\n");
	scanf("%d%d%d", &x, &y, &z);
	while (h1 != start)
	{
		result = result + (h1->cf * pow(x, h1->px) * pow(y, h1->py) * pow(z, h1->pz));
		h1 = h1->link;
	}
	printf("\nPolynomial result is: %f", result);
}

NODE* add_poly(NODE* h1, NODE* h2, NODE* h3)
{
	NODE* p1, *p2;
	int x1, x2, y1, y2, z1, z2, cf1, cf2, cf;
	p1 = h1->link;
	while (p1 != h1)
	{
		x1 = p1->px;
		y1 = p1->py;
		z1 = p1->pz;
		cf1 = p1->cf;

		p2 = h2->link;
		while (p2 != h2)
		{
```

```
                        x2 = p2->px;
                        y2 = p2->py;
                        z2 = p2->pz;
                        cf2 = p2->cf;

                        if (x1 == x2 && y1 == y2 && z1 == z2)
                        {
                                break;
                        }
                        p2 = p2->link;
                }

        if (p2 != h2)
        {
                cf = cf1 + cf2;
                p2->flag = 1;

                if (cf != 0)
                {
                        h3 = insert_rear(cf, x1, y1, z1, h3);
                }
        }
        else
        {
                h3 = insert_rear(cf1, x1, y1, z1, h3);
        }
        p1 = p1->link;
    }

    p2 = h2->link;
    while (p2 != h2)
    {
        if (p2->flag == 0)
        {
                h3 = insert_rear(p2->cf, p2->px, p2->py, p2->pz, h3);
        }
        p2 = p2->link;
    }
    return h3;
}

int main()
{
```

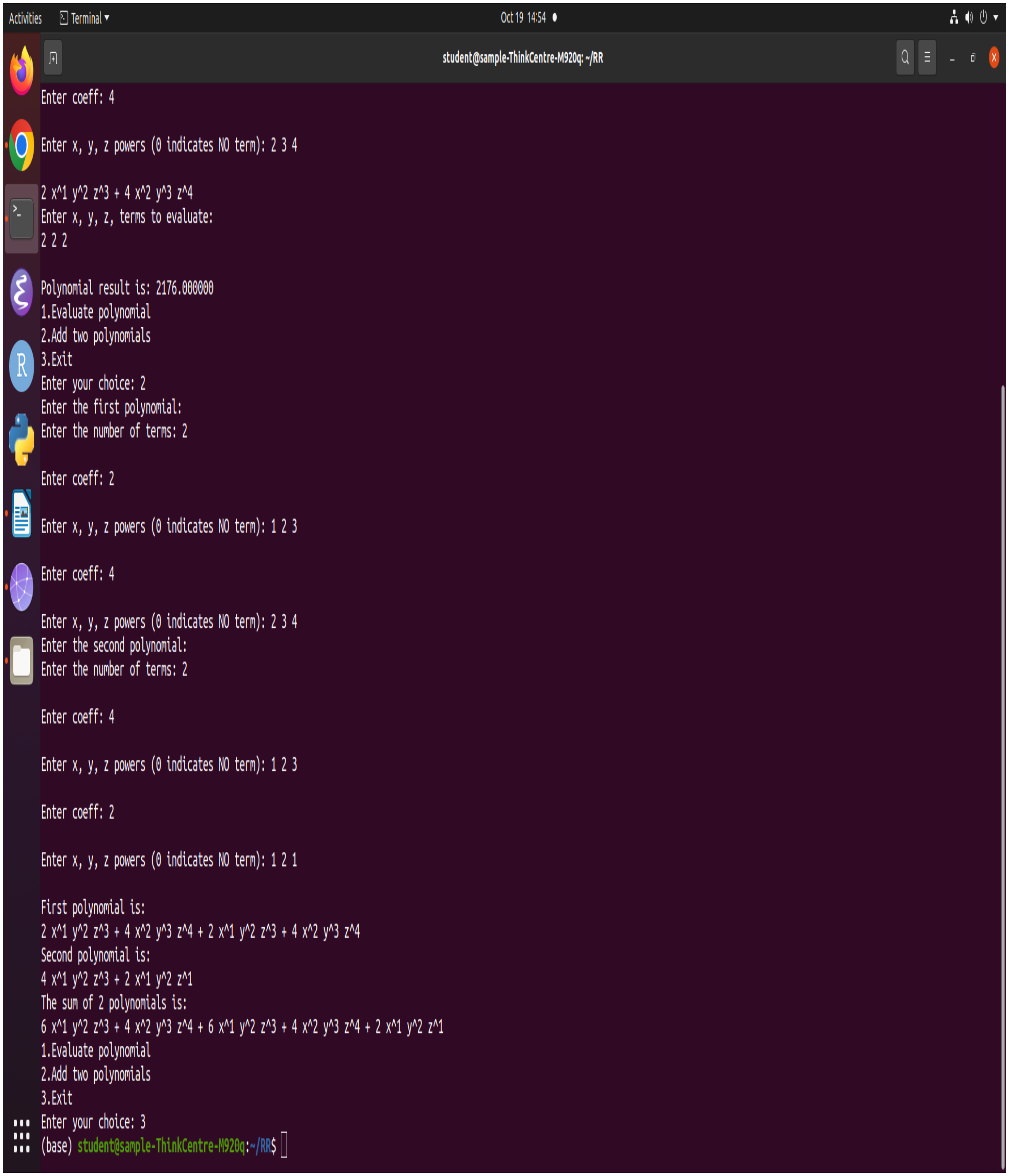
```
NODE* h1, *h2, *h3;
int ch;

h1 = getnode();
h2 = getnode();
h3 = getnode();

h1->link = h1;
h2->link = h2;
h3->link = h3;

while (1)
{
        printf("\n\n1. Evaluate polynomial\n2. Add two polynomials\n3. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &ch);

        switch (ch)
        {
                case 1:
                        printf("\nEnter polynomial to evaluate:\n");
                        h1 = read_poly(h1);
                        display(h1);
                        evaluate(h1);
                        break;

                case 2:
                        printf("\nEnter the first polynomial:");
                        h1 = read_poly(h1);

                        printf("\nEnter the second polynomial:");
                        h2 = read_poly(h2);

                        h3 = add_poly(h1, h2, h3);

                        printf("\nFirst polynomial is: ");
                        display(h1);
                        printf("\nSecond polynomial is: ");
                        display(h2);

                        printf("\nThe sum of 2 polynomials is: ");
                        display(h3);
```

```
                                        break;

                              case 3:
                                        exit(0);
                                        break;
                    }
          }
          return 0;
}
```

Output:



```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 9.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 9.c -lm
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out

1.Evaluate polynomial
2.Add two polynomials
3.Exit
Enter your choice: 1
Enter polynomial to evaluate:

Enter the number of terms: 2

Enter coeff: 2

Enter x, y, z powers (0 indicates NO term): 1 2 3

Enter coeff: 4

Enter x, y, z powers (0 indicates NO term): 2 3 4

2 x^1 y^2 z^3 + 4 x^2 y^3 z^4
Enter x, y, z, terms to evaluate:
2 2 2

Polynomial result is: 2176.000000
1.Evaluate polynomial
2.Add two polynomials
3.Exit
Enter your choice: 2
Enter the first polynomial:
Enter the number of terms: 2

Enter coeff: 2

Enter x, y, z powers (0 indicates NO term): 1 2 3

Enter coeff: 4

Enter x, y, z powers (0 indicates NO term): 2 3 4
Enter the second polynomial:
Enter the number of terms: 2

Enter coeff: 4

Enter x, y, z powers (0 indicates NO term): 1 2 3
```



```
Enter coeff: 4

Enter x, y, z powers (0 indicates NO term): 2 3 4

2 x^1 y^2 z^3 + 4 x^2 y^3 z^4
Enter x, y, z, terms to evaluate:
2 2 2

Polynomial result is: 2176.000000
1.Evaluate polynomial
2.Add two polynomials
3.Exit
Enter your choice: 2
Enter the first polynomial:
Enter the number of terms: 2

Enter coeff: 2

Enter x, y, z powers (0 indicates NO term): 1 2 3

Enter coeff: 4

Enter x, y, z powers (0 indicates NO term): 2 3 4
Enter the second polynomial:
Enter the number of terms: 2

Enter coeff: 4

Enter x, y, z powers (0 indicates NO term): 1 2 3

Enter coeff: 2

Enter x, y, z powers (0 indicates NO term): 1 2 1

First polynomial is:
2 x^1 y^2 z^3 + 4 x^2 y^3 z^4 + 2 x^1 y^2 z^3 + 4 x^2 y^3 z^4
Second polynomial is:
4 x^1 y^2 z^3 + 2 x^1 y^2 z^1
The sum of 2 polynomials is:
6 x^1 y^2 z^3 + 4 x^2 y^3 z^4 + 6 x^1 y^2 z^3 + 4 x^2 y^3 z^4 + 2 x^1 y^2 z^1
1.Evaluate polynomial
2.Add two polynomials
3.Exit
Enter your choice: 3
(base) student@sample-ThinkCentre-M920q:~/RR$
```

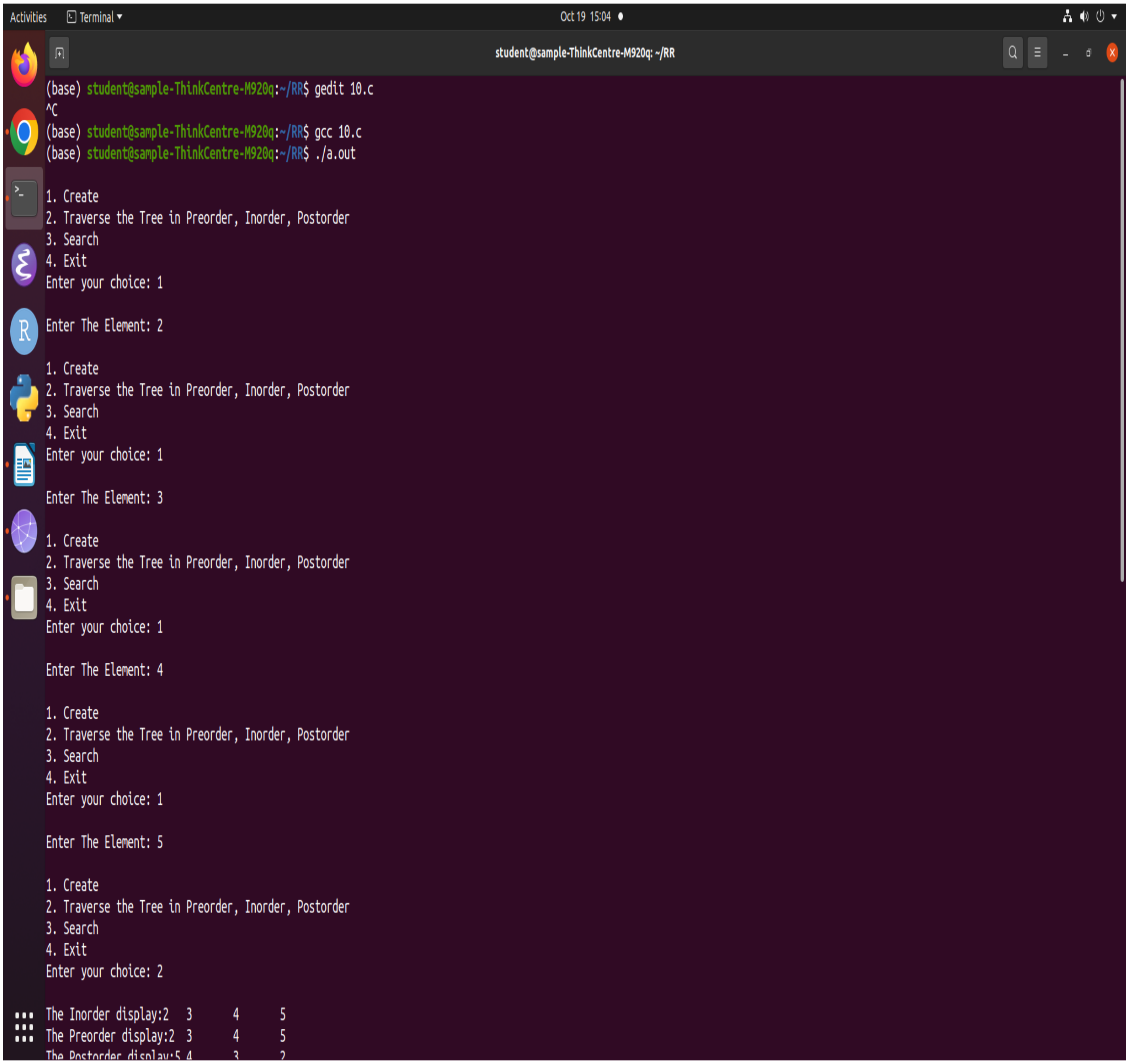
10. Design, Develop and Implement a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers .

a. Create a BST of N Integers: 6, 9, 5, 2, 8, 15, 24, 14, 7, 8, 5, 2
b. Traverse the BST in Inorder, Preorder and Post Order
c. Search the BST for a given element (KEY) and report the appropriate message
d. Exit.

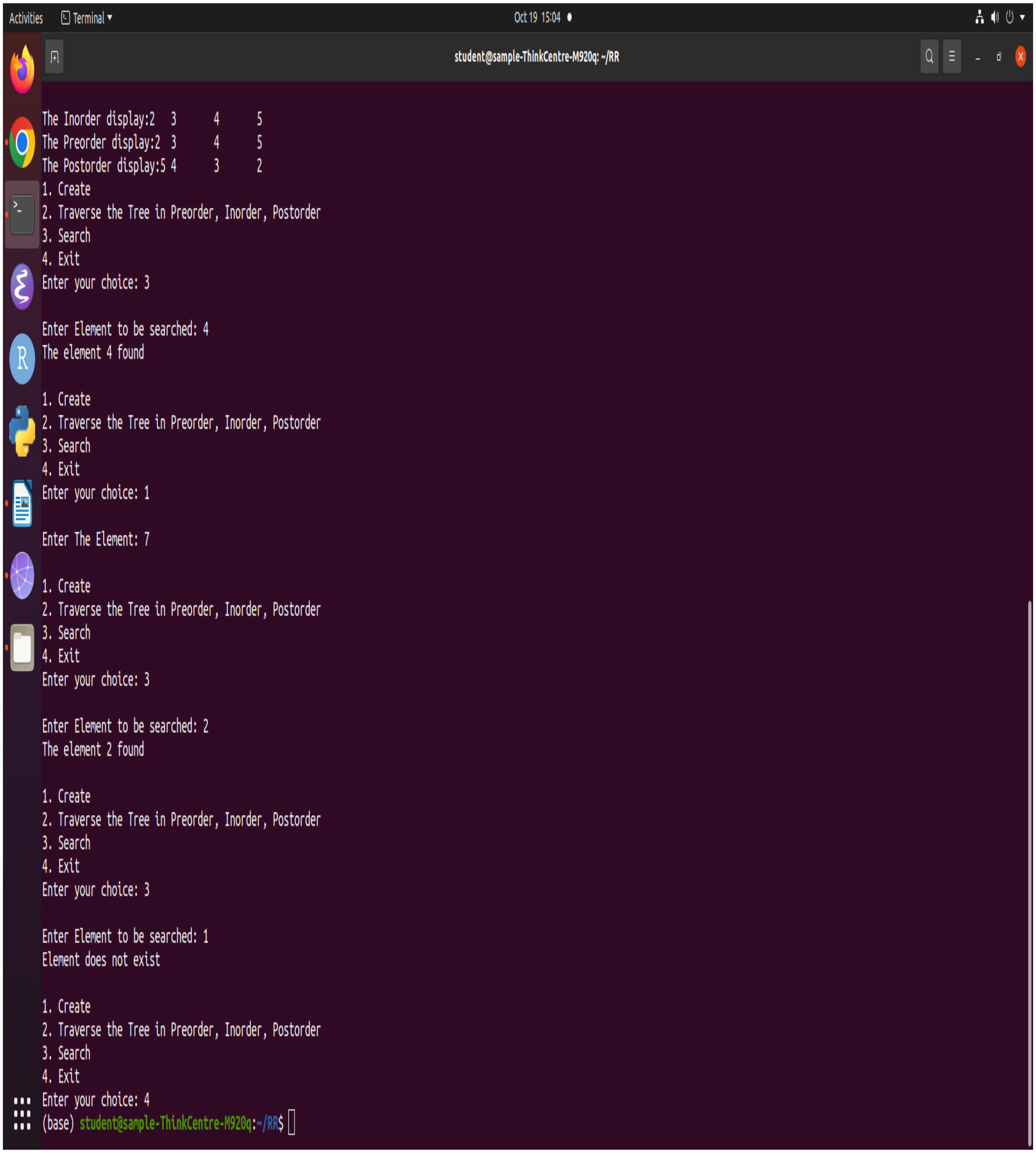
```
#include <stdio.h>
#include <stdlib.h>
struct BST
{
        int info;
        struct BST *llink, *rlink;
};

typedef struct BST *NODE;

NODE insert(NODE root)
{
        NODE temp, cur, prev;
        int item;
        printf("\nEnter The Element: ");
        scanf("%d", &item);
        temp = (NODE)malloc(sizeof(struct BST));
        temp->llink = NULL;
        temp->rlink = NULL;
        temp->info = item;

        if (root == NULL)
                return temp;
        prev = NULL;
        cur = root;
        while (cur != NULL)
        {
                prev = cur;
                if (item < cur->info)
                cur = cur->llink;
                else
                cur = cur->rlink;
        }
        if (item < prev->info)
        prev->llink = temp;
        else
        prev->rlink = temp;
        return root;
}

void inorder(NODE root)
{
        if (root != NULL)
        {
                inorder(root->llink);
                printf("%d\t", root->info);
                inorder(root->rlink);
        }
}

void preorder(NODE root)
{
        if (root != NULL)
        {
                printf("%d\t", root->info);
                preorder(root->llink);
                preorder(root->rlink);
        }
}

void postorder(NODE root)
{
        if (root != NULL)
        {
                postorder(root->llink);
                postorder(root->rlink);
                printf("%d\t", root->info);
        }
}

NODE search(NODE root, int key)
{
        if (root == NULL)
        return NULL;
        if (root->info == key)
        return root;
        if (key < root->info)
        return search(root->llink, key);
        else
        return search(root->rlink, key);
}

int main()
{
        int choice, key;
        NODE root = NULL;
        while (1)
        {
                printf("\n1. Create");
                printf("\n2. Traverse the Tree in Preorder, Inorder, Postorder");
                printf("\n3. Search");
                printf("\n4. Exit");
                printf("\nEnter your choice: ");
                scanf("%d", &choice);

                switch (choice)
                {
                        case 1:
                                root = insert(root);
                                break;

                        case 2:
                                if (root == NULL)
                                printf("Tree Is Not Created\n");
                                else
                                {
                                        printf("\nThe Inorder display: ");
                                        inorder(root);
                                        printf("\nThe Preorder display: ");
                                        preorder(root);
                                        printf("\nThe Postorder display: ");
                                        postorder(root);
                                }
                                break;
                        case 3:
                                printf("\nEnter Element to be searched: ");
                                scanf("%d", &key);
                                NODE temp = search(root, key);
                                if (temp == NULL)
                                printf("Element does not exist\n");
                                else
```

```
                                        printf("The element %d found\n", temp->info);
                              break;
                              default:
                              exit(0);
                    }
          }
          return 0;
}
```

Output:



```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 10.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 10.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out

1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 1

Enter The Element: 2

1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 1

Enter The Element: 3

1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 1

Enter The Element: 4

1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 1

Enter The Element: 5

1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 2

The Inorder display:2   3       4       5
The Preorder display:2  3       4       5
The Postorder display:5 4       3       2
```


Activities Terminal Oct 19 15:04
student@sample-ThinkCentre-M920q: ~/RR

```
The Inorder display:2   3       4       5
The Preorder display:2  3       4       5
The Postorder display:5 4       3       2
1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 3

Enter Element to be searched: 4
The element 4 found

1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 1

Enter The Element: 7

1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 3

Enter Element to be searched: 2
The element 2 found

1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 3

Enter Element to be searched: 1
Element does not exist

1. Create
2. Traverse the Tree in Preorder, Inorder, Postorder
3. Search
4. Exit
Enter your choice: 4
(base) student@sample-ThinkCentre-M920q:~/RR$
```

11.Design, Develop and Implement a Program in C for the following operations on Graph(G) of Cities

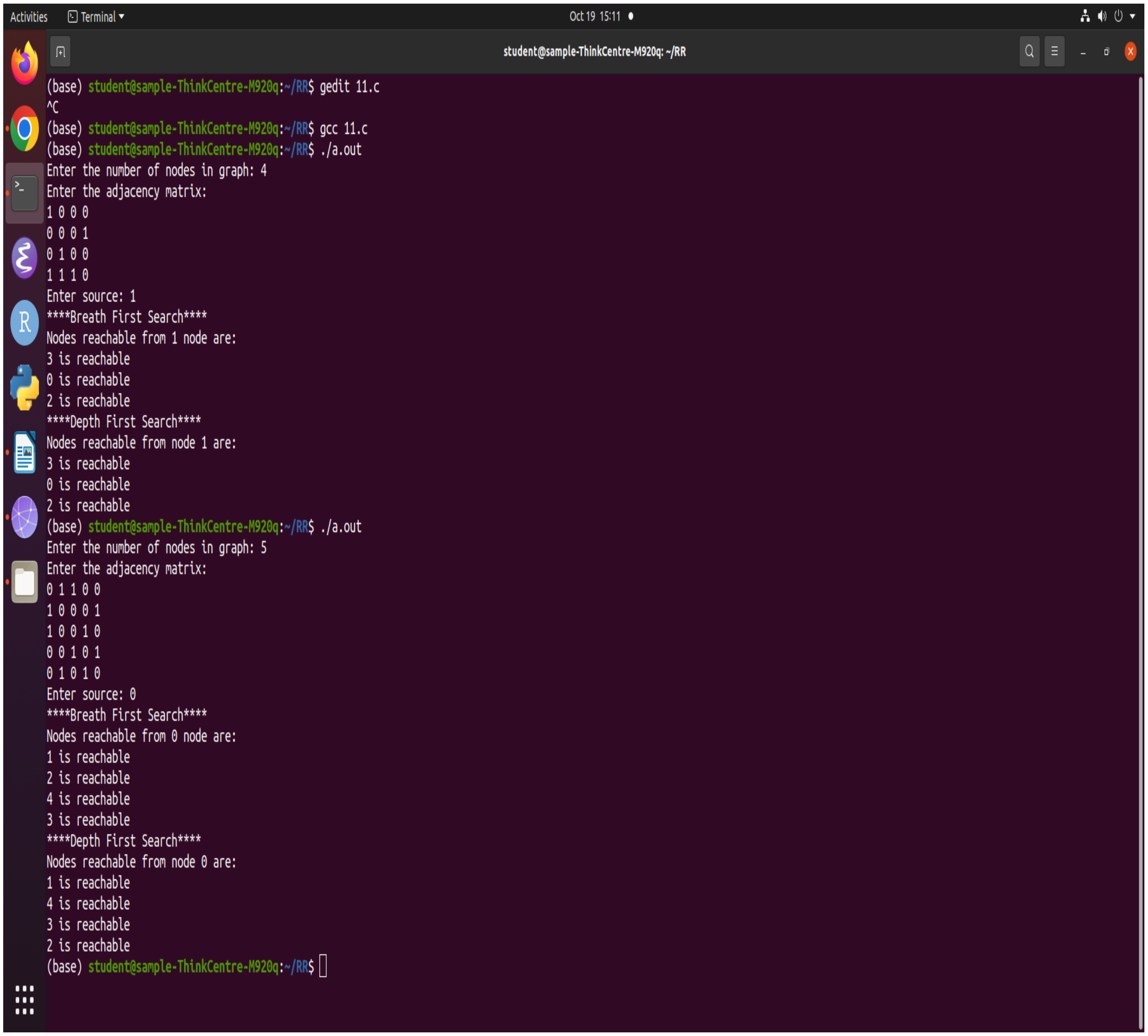
a. Create a Graph of N cities using Adjacency Matrix.

b. Print all the nodes reachable from a given starting node in a digraph using BFS/DFS method

```
#include <stdio.h>

int n, a[10][10], src;
int visited[10];
int q[10];

void bfs(int src)
{
	int v, front = 0, rear = -1;
	visited[src] = 1;
	q[++rear] = src;
	while (front <= rear)
	{
		int u = q[front++];
		for (v = 0; v < n; v++)
		{
			if (a[u][v] == 1 && visited[v] == 0)
			{
				visited[v] = 1;
				q[++rear] = v;
				printf("%d is reachable\n", v);
			}
		}
	}
}

void dfs(int u)
{
	int v;
	visited[u] = 1;
	for (v = 0; v < n; v++)
	{
		if (a[u][v] == 1 && visited[v] == 0)
		{
			printf("%d is reachable\n", v);
```

```
                        dfs(v);
                }
        }
}

int main()
{
        int i, j;
        printf("Enter the number of nodes in graph: ");
        scanf("%d", &n);
        printf("Enter the adjacency matrix:\n");

        for (i = 0; i < n; i++)
        {
                for (j = 0; j < n; j++)
                {
                        scanf("%d", &a[i][j]);
                }
        }

        printf("Enter source: ");
        scanf("%d", &src);

        printf("****Breath First Search****\n");
        printf("Nodes reachable from %d node are:\n", src);

        for (i = 0; i < n; i++)
        {
                visited[i] = 0;
        }
        bfs(src);

        printf("****Depth First Search****\n");
        printf("Nodes reachable from node %d are:\n", src);

        for (i = 0; i < n; i++)
        {
                visited[i] = 0;
        }
        dfs(src);
        return 0;
}
```

Output:



```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 11.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 11.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the number of nodes in graph: 4
Enter the adjacency matrix:
1 0 0 0
0 0 0 1
0 1 0 0
1 1 1 0
Enter source: 1
****Breath First Search****
Nodes reachable from 1 node are:
3 is reachable
0 is reachable
2 is reachable
****Depth First Search****
Nodes reachable from node 1 are:
3 is reachable
0 is reachable
2 is reachable
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
Enter the number of nodes in graph: 5
Enter the adjacency matrix:
0 1 1 0 0
1 0 0 0 1
1 0 0 1 0
0 0 1 0 1
0 1 0 1 0
Enter source: 0
****Breath First Search****
Nodes reachable from 0 node are:
1 is reachable
2 is reachable
4 is reachable
3 is reachable
****Depth First Search****
Nodes reachable from node 0 are:
1 is reachable
4 is reachable
3 is reachable
2 is reachable
(base) student@sample-ThinkCentre-M920q:~/RR$ 
```

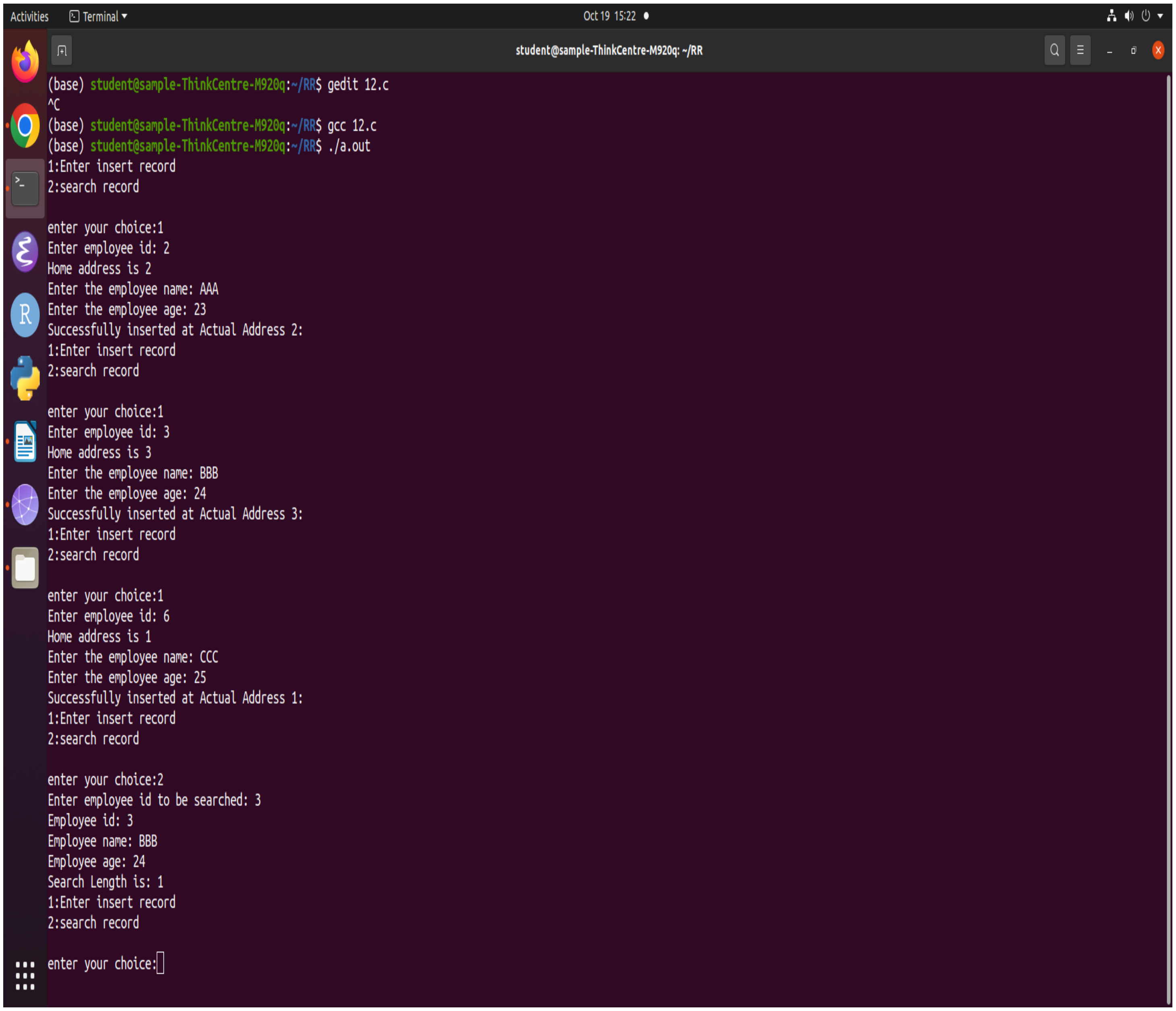
12. Given a File of N employee records with a set K of Keys (4-digit) which uniquely determine the records in file F. Assume that file F is maintained in memory by a Hash Table (HT) of m memory locations with L as the set of memory addresses (2-digit) of locations in HT. Let the keys in K and addresses in L are Integers. Design and develop a Program in C that uses Hash function H: K --> L as H(K)=K mod m (remainder method), and implement hashing technique to map a given key K to the address space L. Resolve the collision (if any) using linear probing.

```
#include <stdio.h>

#define MAX_ADDR 5

struct employee
{
int id, age;
char name[25];
} emp[MAX_ADDR];

int hash(int key)
{
	return key % MAX_ADDR;
}

int main(void)
{
	int i, ch, count = 0, index, haddr, id;
	for (;;)
	{
		printf("Enter 1 to insert record\n2 to search record\n");
		scanf("%d", &ch);
		switch (ch)
		{
			case 1:
				if (count == MAX_ADDR)
			{
				printf("No free address space\n");
				break;
			}
			printf("Enter employee id: ");
			scanf("%d", &id);
			haddr = hash(id);
```

```
            printf("Home address is %d\n", haddr);

            for (i = 0; i < MAX_ADDR; i++)
            {
                index = (haddr + i) % MAX_ADDR;
                if (emp[index].id == 0)
                {
                    emp[index].id = id;
                    printf("Enter the employee name: ");
                    scanf("%s", emp[index].name);
                    printf("Enter the employee age: ");
                    scanf("%d", &emp[index].age);
                    count++;
                    printf("Successfully inserted at Actual Address %d:\n", index);
                    break;
                }
            }
            break;

            case 2:
                printf("Enter employee id to be searched: ");
                scanf("%d", &id);
                haddr = hash(id);

                for (i = 0; i < MAX_ADDR; i++)
                {
                    index = (haddr + i) % MAX_ADDR;

                    if (emp[index].id == 0)
                {
                    printf("Key not present\n");
                    break;
                } else if (emp[index].id == id)
                {
                    printf("Employee id: %d\n", emp[index].id);
                    printf("Employee name: %s\n", emp[index].name);
                    printf("Employee age: %d\n", emp[index].age);
                    printf("Search Length is: %d\n", i + 1);
                    break;
                }
        }

        break;
```

```
                        default:
                        return 0;
                }
        }
}
```

Output:



```
(base) student@sample-ThinkCentre-M920q:~/RR$ gedit 12.c
^C
(base) student@sample-ThinkCentre-M920q:~/RR$ gcc 12.c
(base) student@sample-ThinkCentre-M920q:~/RR$ ./a.out
1:Enter insert record
2:search record

enter your choice:1
Enter employee id: 2
Home address is 2
Enter the employee name: AAA
Enter the employee age: 23
Successfully inserted at Actual Address 2:
1:Enter insert record
2:search record

enter your choice:1
Enter employee id: 3
Home address is 3
Enter the employee name: BBB
Enter the employee age: 24
Successfully inserted at Actual Address 3:
1:Enter insert record
2:search record

enter your choice:1
Enter employee id: 6
Home address is 1
Enter the employee name: CCC
Enter the employee age: 25
Successfully inserted at Actual Address 1:
1:Enter insert record
2:search record

enter your choice:2
Enter employee id to be searched: 3
Employee id: 3
Employee name: BBB
Employee age: 24
Search Length is: 1
1:Enter insert record
2:search record

enter your choice:
```

ATRIA INSTITUTE OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VISION

TO BE A MODEL CENTER FOR EDUCATION AND HIGHER LEARNING TO MEET THE COMPUTING CHALLENGES OF THE INDUSTRIAL DEMANDS AND SOCIETAL NEEDS.

MISSION

M1- EMPOWER THE GRADUATES WITH THE FUNDAMENTALS IN DESIGN AND IMPLEMENTATION OF COMPUTATIONAL SYSTEMS THROUGH CURRICULUM ANDRESEARCH IN COLLABORATION WITH INDUSTRIES AND INSTITUTES OF REPUTE.

M2- TO DEVELOP A STATE-OF-THE-ART INFRASTRUCTURE AND CREATE AMBIENCE (ENVIRONMENT) CAPABLE OF INTERDISCIPLINARY RESEARCH AND SKILL ENHANCEMENT.

M3- TO NURTURE FACULTY WHO HAVE ACADEMIC AND INDUSTRY EXPOSURE, TO IMPART dOMAIN KNOWLEDGE AND TO POSITION OUR STUDENTS IN THE GLOBAL IT ECOSYSTEM.

M4- TO CARRY OUT PROFESSIONAL BRILLIANCE WITH ETHICAL AND MORAL STANDARDS