Reg. No.: 21BCB0117
Name: Sankhayan Bhattacharjee

Q1. Stack applications
i) Infix to Postfix Conversion
ii) Evaluation of Postfix expression
iii) Balancing Symbols

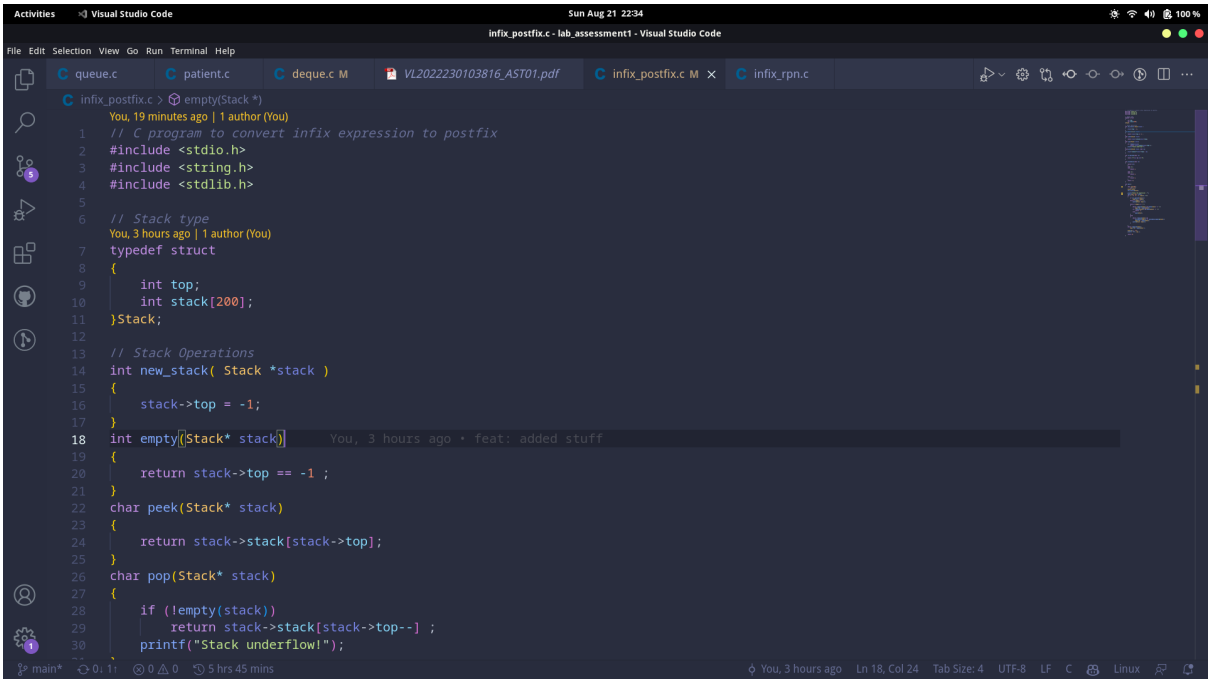
i) Infix to Postfix Conversion
Program Code:

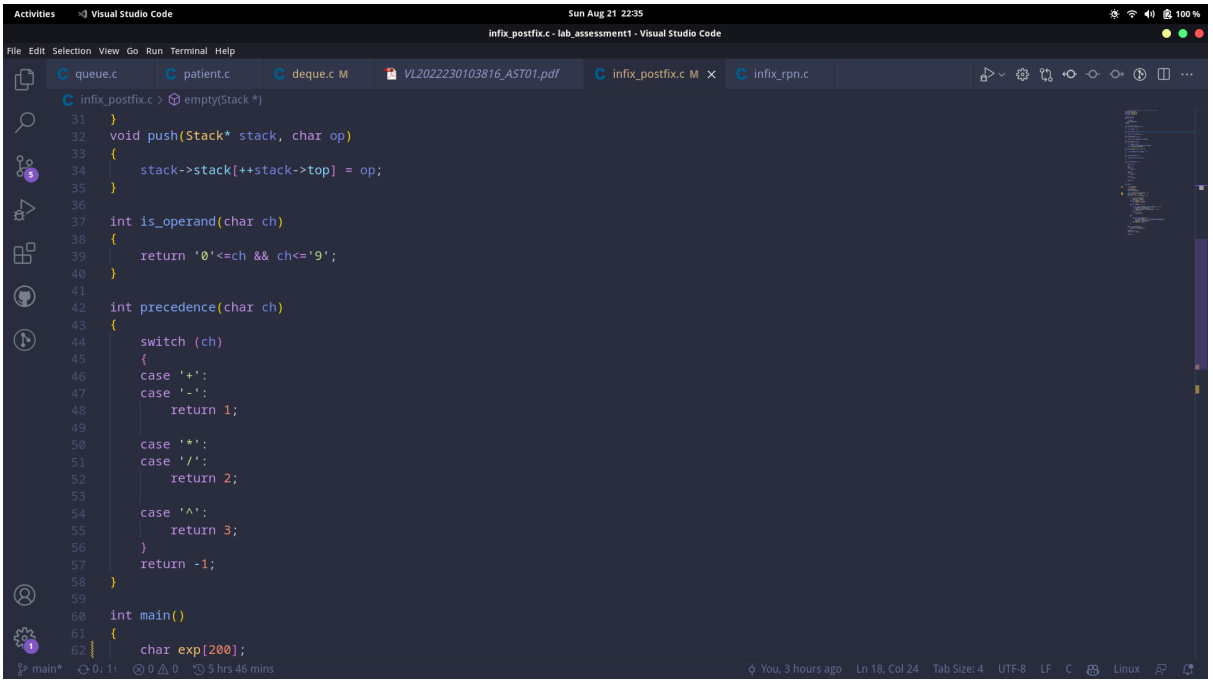
```c
// C program to convert infix expression to postfix
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

// Stack type
typedef struct
{
   int top;
   int stack[200];
}Stack;

// Stack Operations
int new_stack( Stack *stack )
{
   stack->top = -1;
}
int empty(Stack* stack)
{
   return stack->top == -1 ;
}
char peek(Stack* stack)
{
   return stack->stack[stack->top];
}
char pop(Stack* stack)
{
   if (!empty(stack))
       return stack->stack[stack->top--] ;
   printf("Stack underflow!");
}
void push(Stack* stack, char op)
{
   stack->stack[++stack->top] = op;
}

int is_operand(char ch)
{
   return '0'<=ch && ch<='9';
}

int precedence(char ch)
```

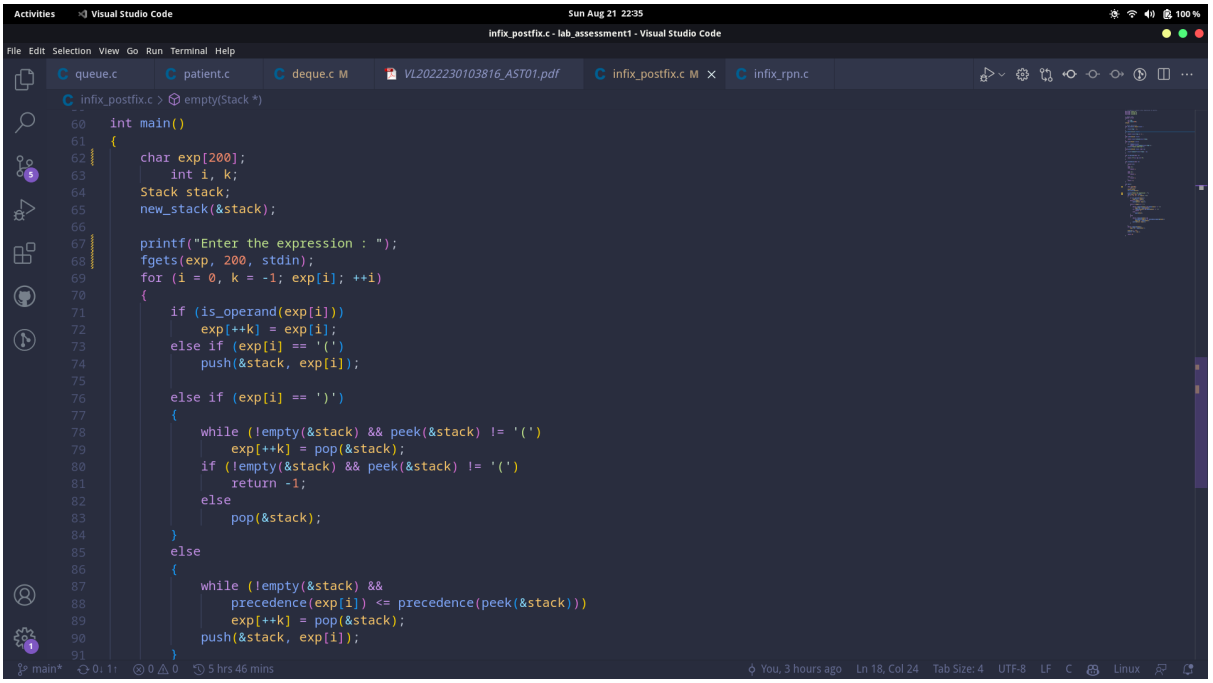
```
{
    switch (ch)
    {
    case '+':
    case '-':
        return 1;

    case '*':
    case '/':
        return 2;

    case '^':
        return 3;
    }
    return -1;
}

int main()
{
    char exp[200];
    int i, k;
    Stack stack;
    new_stack(&stack);
    printf("Enter the expression : ");
    fgets(exp, 200, stdin);
    for (i = 0, k = -1; exp[i]; ++i)
    {
        if (is_operand(exp[i]))
            exp[++k] = exp[i];
        else if (exp[i] == '(')
            push(&stack, exp[i]);

        else if (exp[i] == ')')
        {
            while (!empty(&stack) && peek(&stack) != '(')
                exp[++k] = pop(&stack);
            if (!empty(&stack) && peek(&stack) != '(')
                return -1;
            else
                pop(&stack);
        }
        else
        {
            while (!empty(&stack) &&
                precedence(exp[i]) <= precedence(peek(&stack)))
                exp[++k] = pop(&stack);
            push(&stack, exp[i]);
        }

    }
    while (!empty(&stack))
```

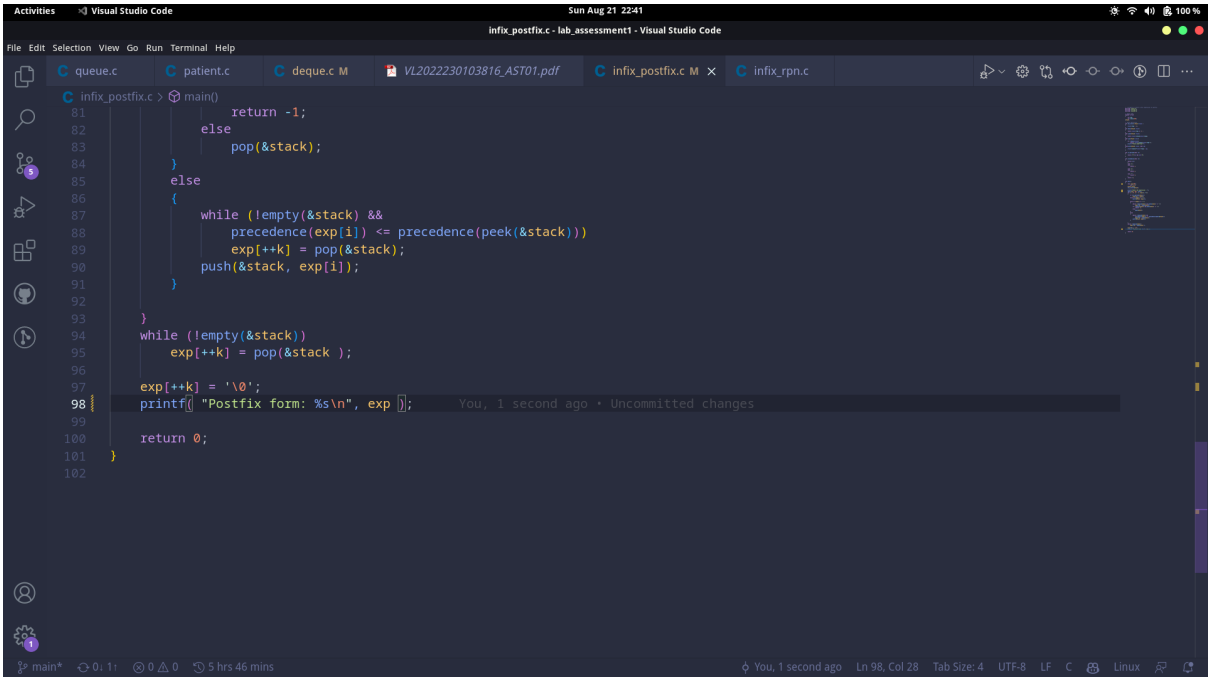
```
        exp[++k] = pop(&stack );

    exp[++k] = '\0';
    printf( "%s", exp );

    return 0;
}
```

Program Screenshots:









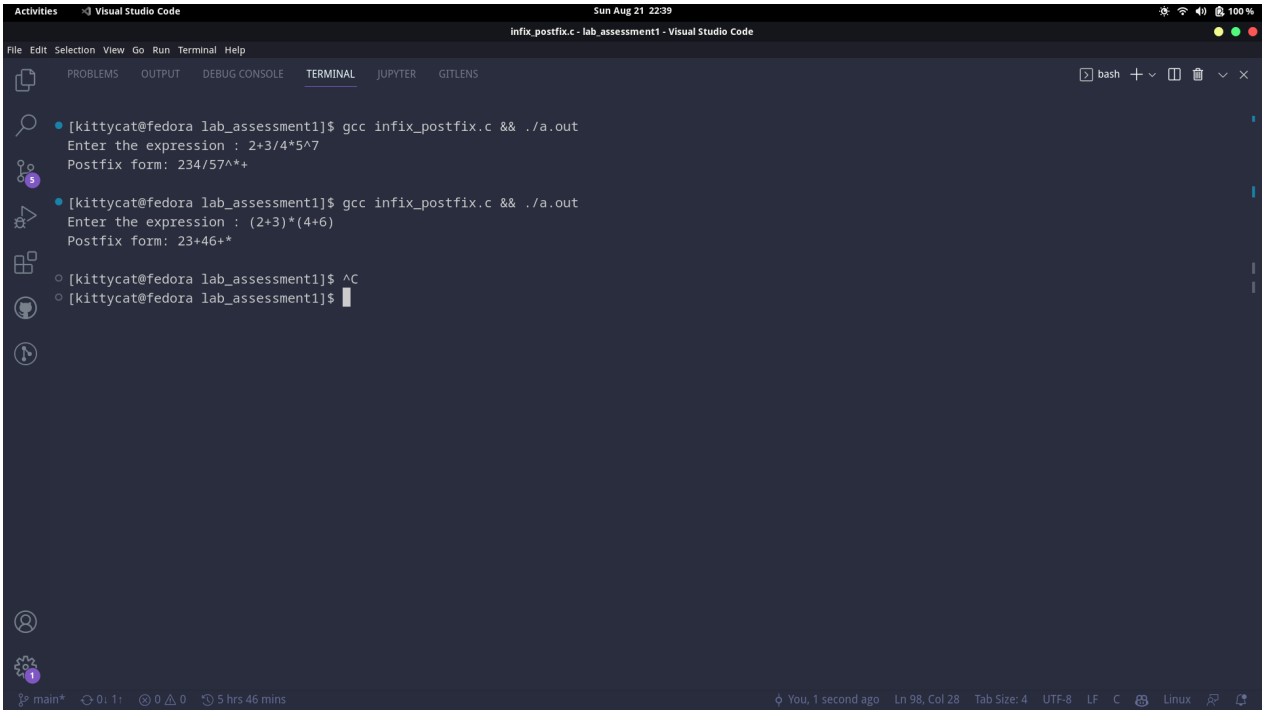
Output:

[kittycat@fedora lab_assessment1]$ gcc infix_postfix.c && ./a.out
Enter the expression : 2+3/4*5^7
Postfix form: 234/57^*+

[kittycat@fedora lab_assessment1]$ gcc infix_postfix.c && ./a.out
Enter the expression : (2+3)*(4+6)
Postfix form: 23+46+*

[kittycat@fedora lab_assessment1]$

Output Screenshot:



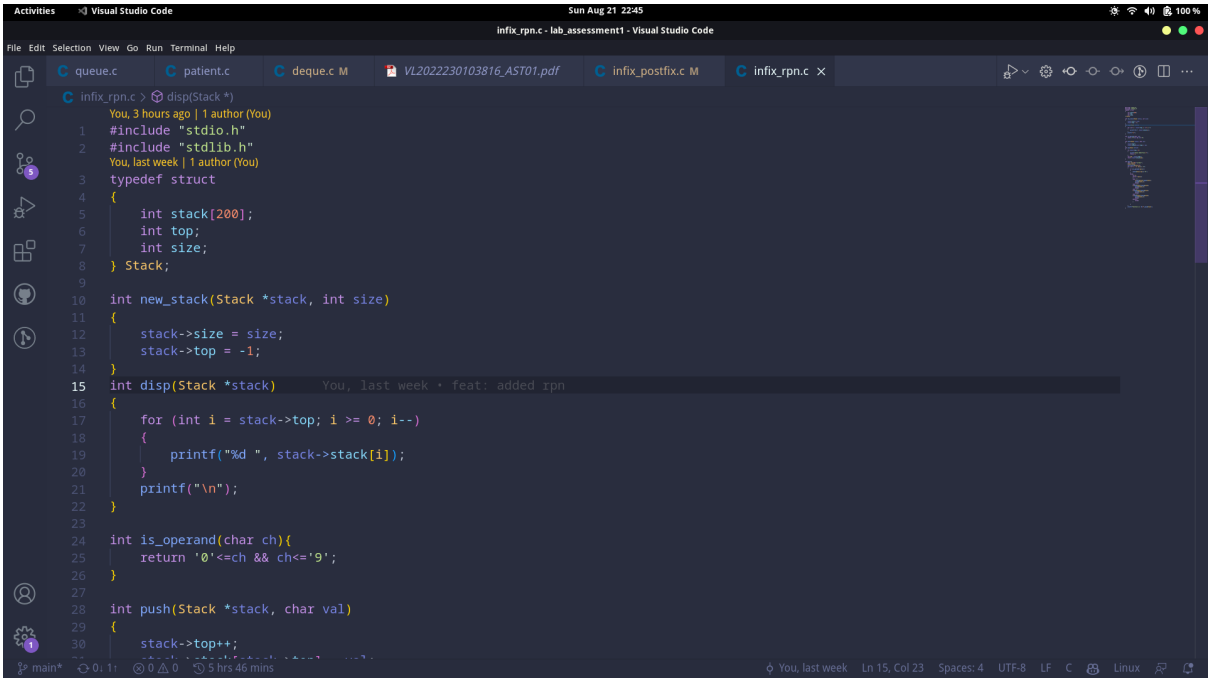
ii) Evaluation of Postfix expression:

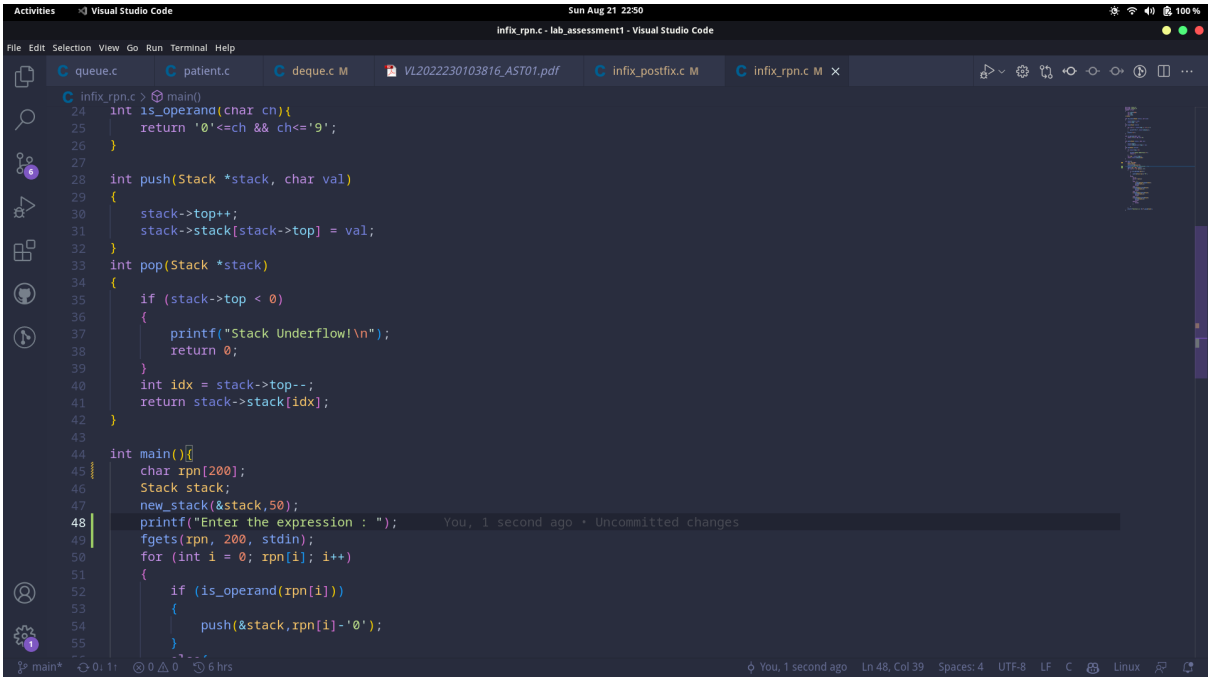
Program Code:

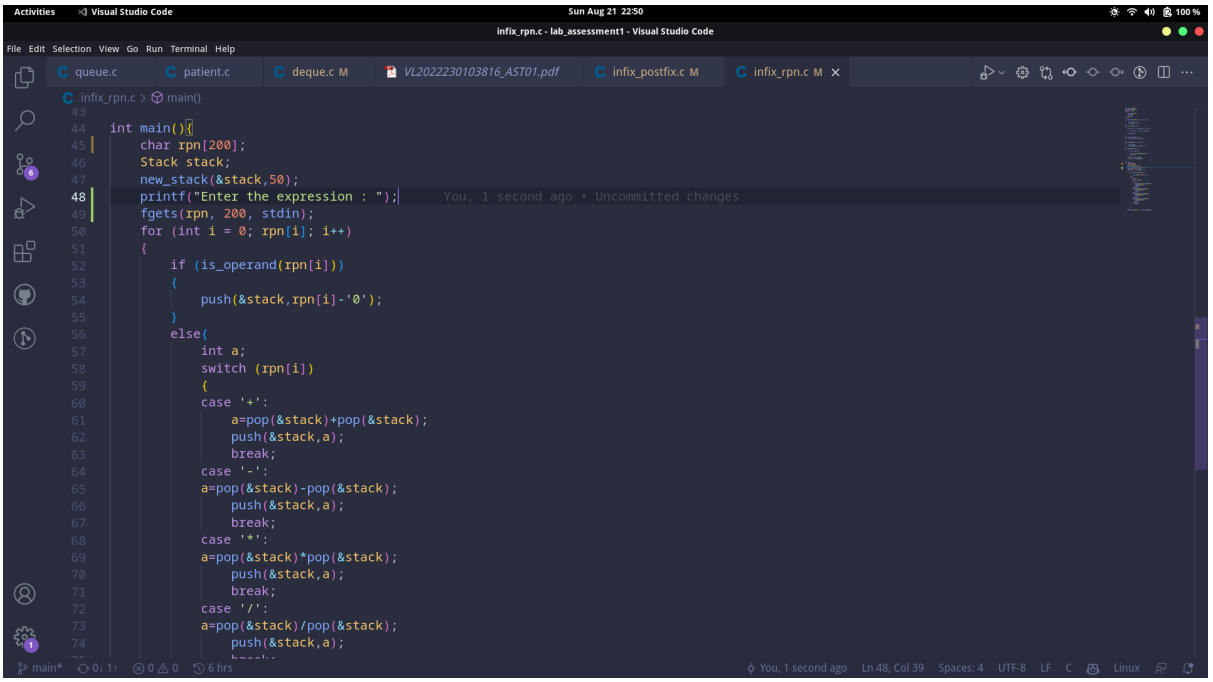
```c
#include "stdio.h"
#include "stdlib.h"
typedef struct
{
    int stack[200];
    int top;
    int size;
} Stack;

int new_stack(Stack *stack, int size)
{
    stack->size = size;
    stack->top = -1;
}
int disp(Stack *stack)
{
    for (int i = stack->top; i >= 0; i--)
    {
        printf("%d ", stack->stack[i]);
    }
    printf("\n");
}

int is_operand(char ch){
    return '0'<=ch && ch<='9';
}

int push(Stack *stack, char val)
{
    stack->top++;
    stack->stack[stack->top] = val;
```

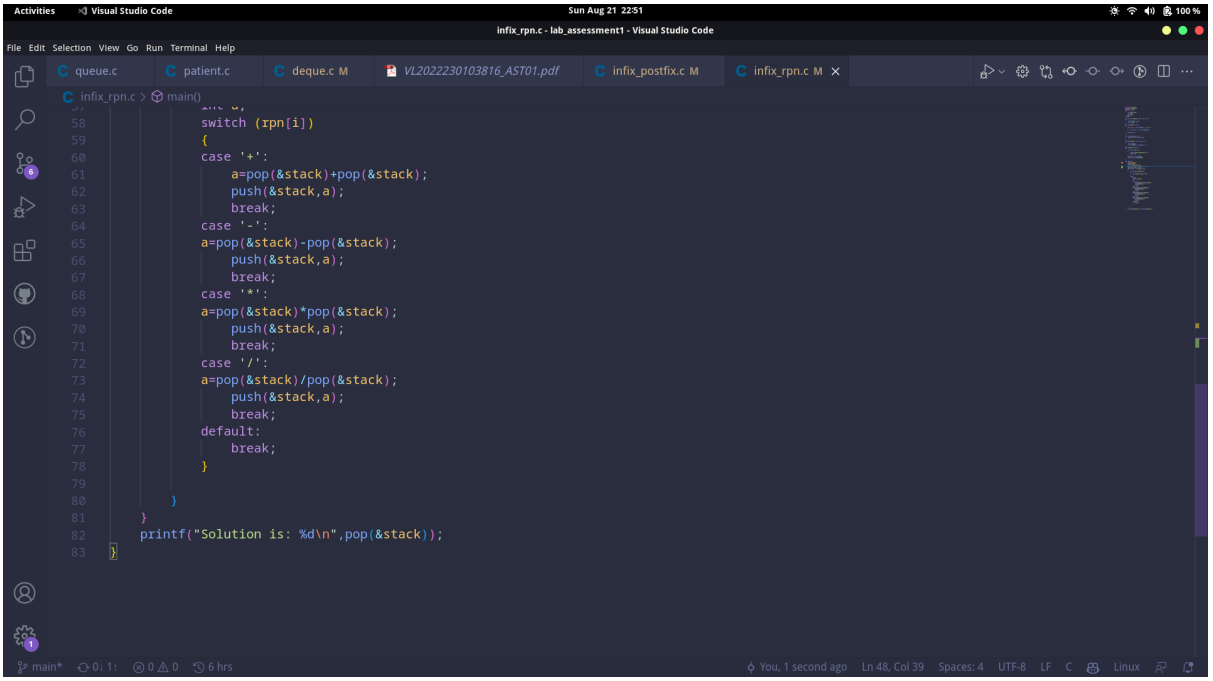
```
}
int pop(Stack *stack)
{
    if (stack->top < 0)
    {
        printf("Stack Underflow!\n");
        return 0;
    }
    int idx = stack->top--;
    return stack->stack[idx];
}
int main(){
    char rpn[200];
    Stack stack;
    new_stack(&stack,50);
    printf("Enter the expression : ");
    for (int i = 0; rpn[i]; i++)
    {
        if (is_operand(rpn[i]))
        {
            push(&stack,rpn[i]-'0');
        }
        else{
            int a;
            switch (rpn[i])
            {
            case '+':
                a=pop(&stack)+pop(&stack);
                push(&stack,a);
                break;
            case '-':
            a=pop(&stack)-pop(&stack);
                push(&stack,a);
                break;
            case '*':
            a=pop(&stack)*pop(&stack);
                push(&stack,a);
                break;
            case '/':
            a=pop(&stack)/pop(&stack);
                push(&stack,a);
                break;
            default:
                break;
            }
        }
    }
    printf("Solution is: %d\n",pop(&stack));
}
```

Program Screenshots:





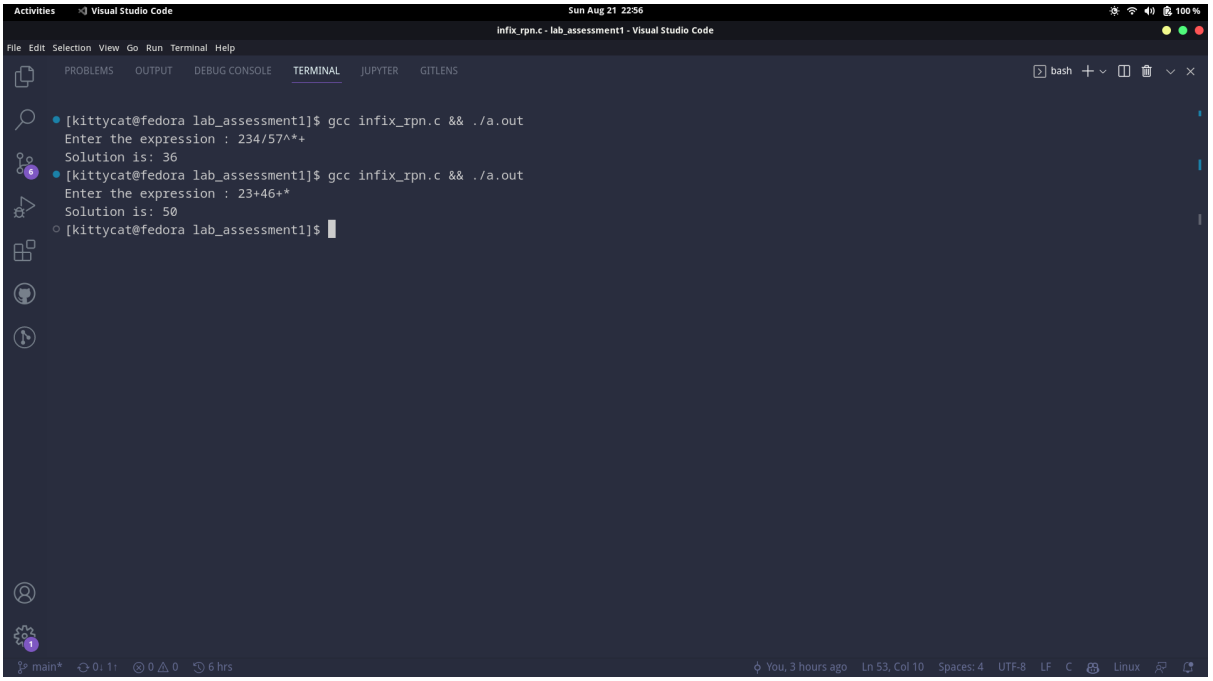




Output:

[kittycat@fedora lab_assessment1]$ gcc infix_rpn.c && ./a.out
Enter the expression : 234/57^*+
Solution is: 36
[kittycat@fedora lab_assessment1]$ gcc infix_rpn.c && ./a.out
Enter the expression : 23+46+*
Solution is: 50
[kittycat@fedora lab_assessment1]$

Output Screenshot:



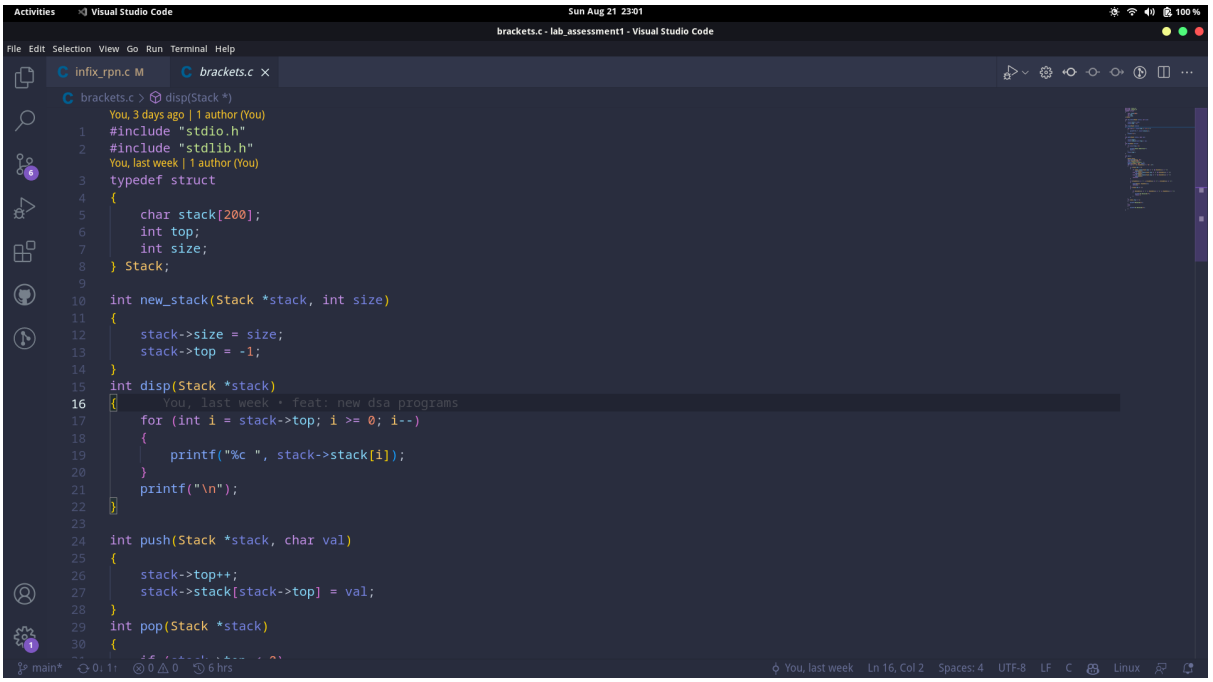
iii) Balancing Symbols

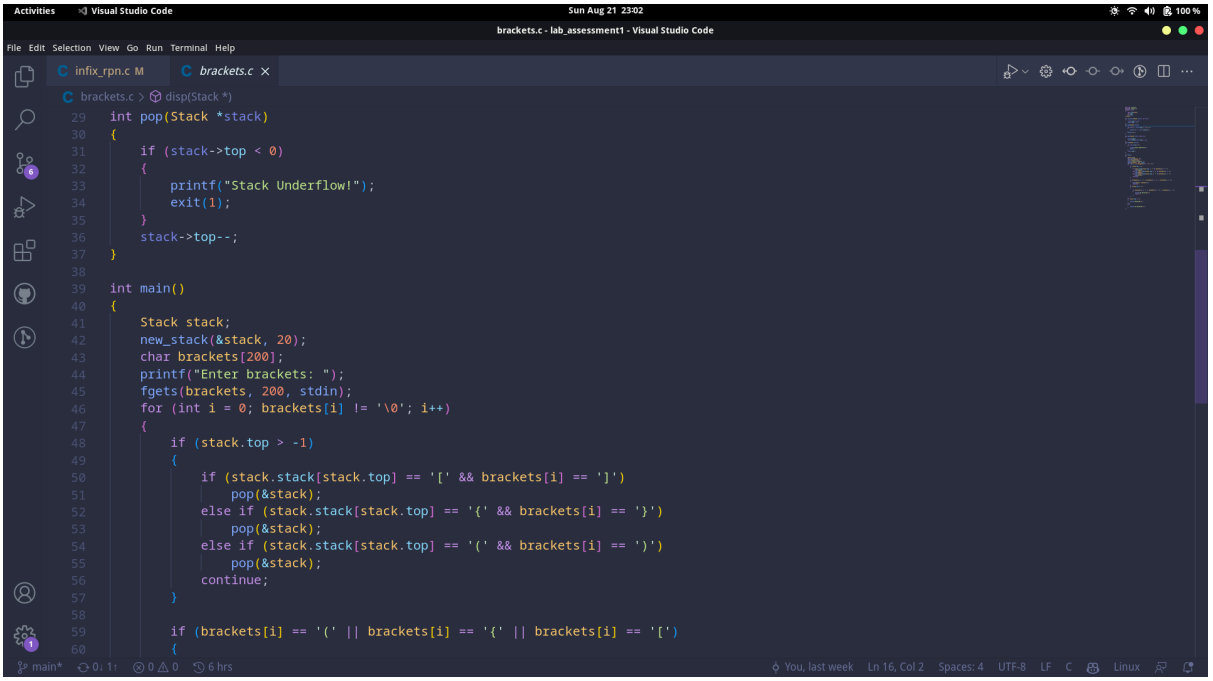
Program Code:

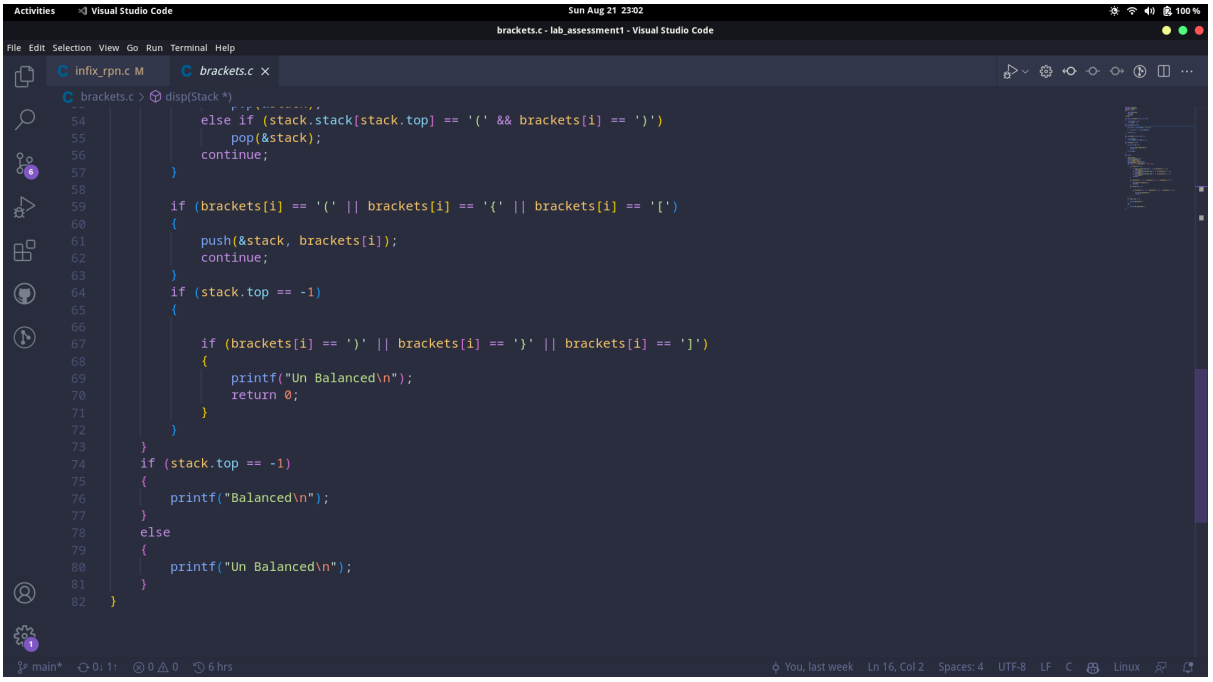
```
#include "stdio.h"
#include "stdlib.h"
typedef struct
{
   char stack[200];
   int top;
   int size;
} Stack;

int new_stack(Stack *stack, int size)
{
   stack->size = size;
   stack->top = -1;
}
int disp(Stack *stack)
{
   for (int i = stack->top; i >= 0; i--)
   {
       printf("%c ", stack->stack[i]);
   }
   printf("\n");
}

int push(Stack *stack, char val)
{
   stack->top++;
   stack->stack[stack->top] = val;
}
int pop(Stack *stack)
{
   if (stack->top < 0)
   {
       printf("Stack Underflow!");
       exit(1);
   }
   stack->top--;
}

int main()
{
   Stack stack;
   new_stack(&stack, 20);
   char brackets[200];
   printf("Enter brackets: ");
   fgets(brackets, 200, stdin);
   for (int i = 0; brackets[i] != '\0'; i++)
   {
       if (stack.top > -1)
```

```
        {
          if (stack.stack[stack.top] == '[' && brackets[i] == ']')
                pop(&stack);
          else if (stack.stack[stack.top] == '{' && brackets[i] == '}')
                pop(&stack);
          else if (stack.stack[stack.top] == '(' && brackets[i] == ')')
                pop(&stack);
            continue;
        }

      if (brackets[i] == '(' || brackets[i] == '{' || brackets[i] =='[')
        {
            push(&stack, brackets[i]);
            continue;
        }
        if (stack.top == -1)
        {

     if (brackets[i] == ')' || brackets[i] == '}' || brackets[i] == ']')
            {
                printf("Un Balanced\n");
                return 0;
            }
        }
    }
    if (stack.top == -1)
    {
        printf("Balanced\n");
    }
    else
    {
        printf("Un Balanced\n");
    }
}
```

Program Screenshots:



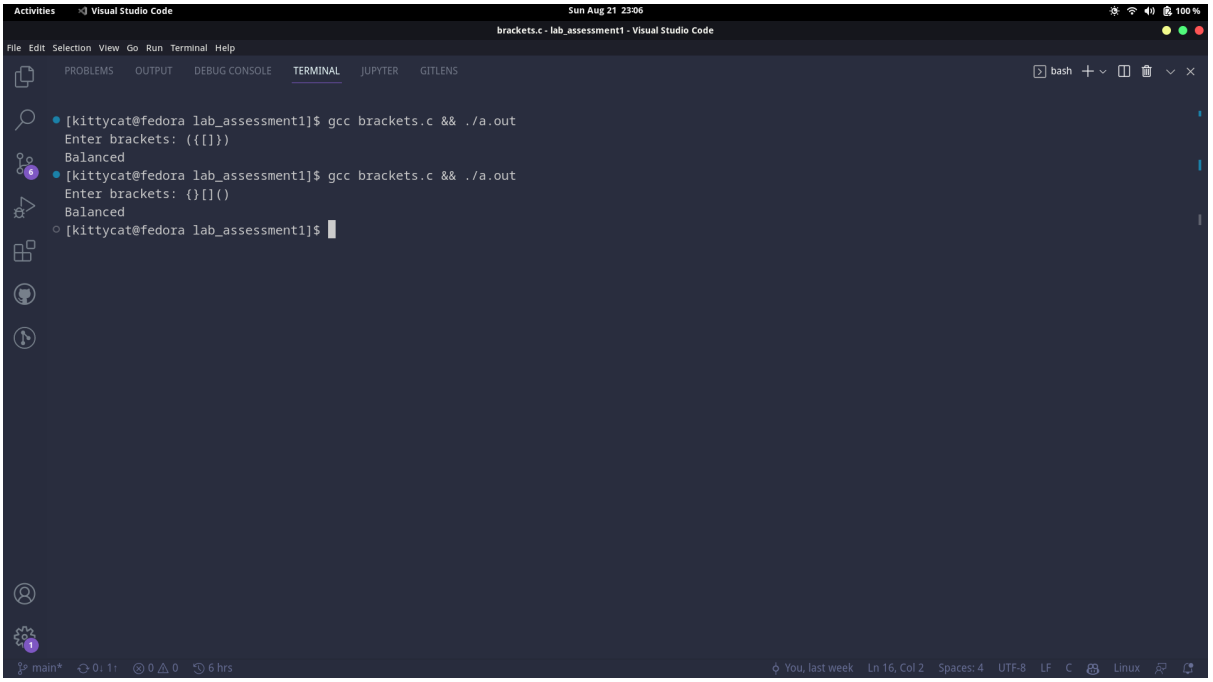




Output:

```
[kittycat@fedora lab_assessment1]$ gcc brackets.c && ./a.out
Enter brackets: ({[]})
Balanced
[kittycat@fedora lab_assessment1]$ gcc brackets.c && ./a.out
Enter brackets: {}[]()
Balanced
[kittycat@fedora lab_assessment1]$
```

Output Screenshots:



2.Write a menu driven program to perform static implementation of a queue data structure with all possible functions.

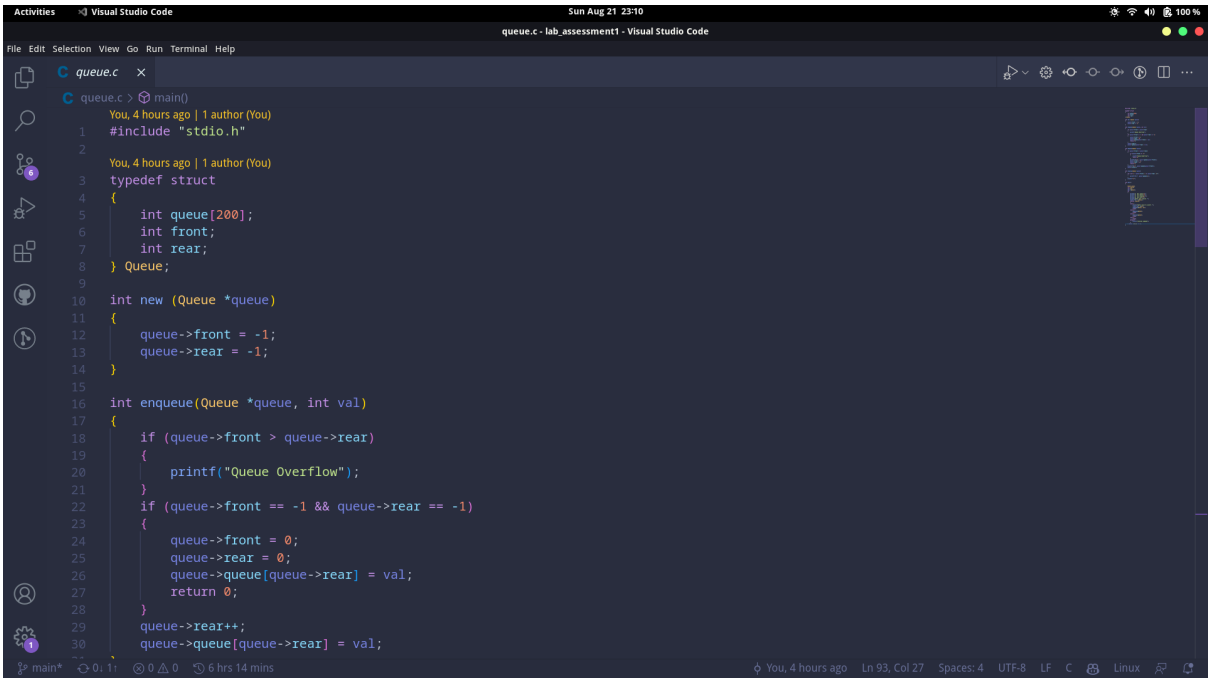
Program Code:

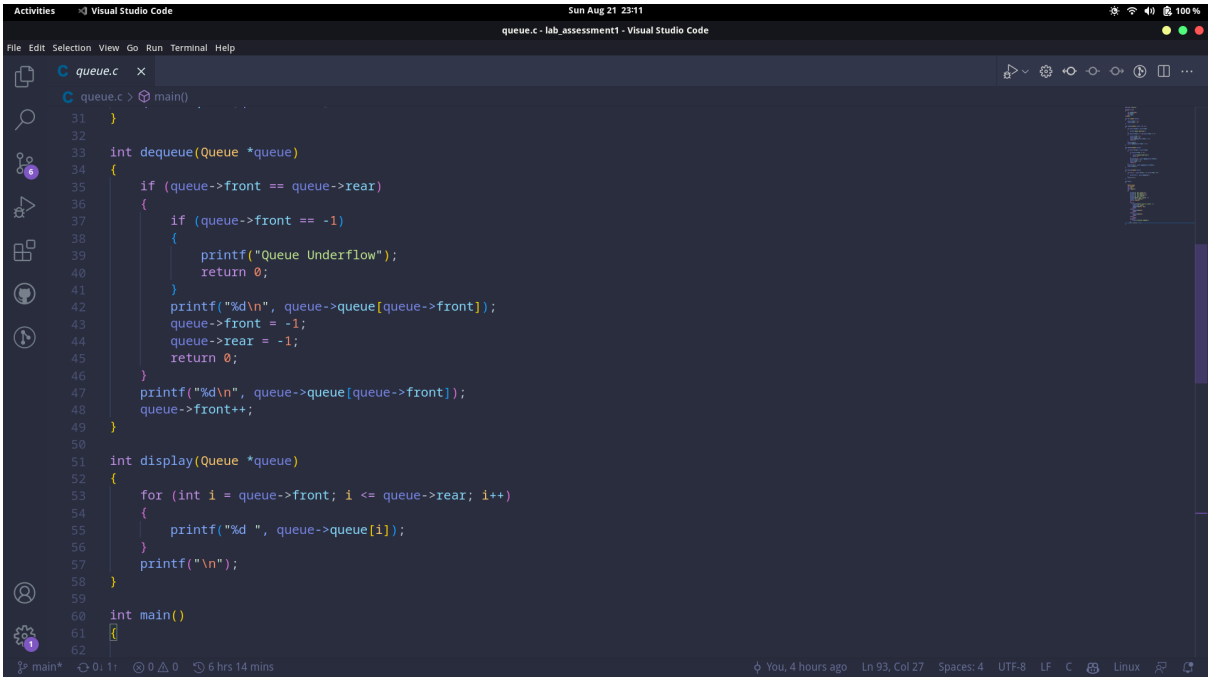
```c
#include "stdio.h"

typedef struct
{
    int queue[200];
    int front;
    int rear;
} Queue;

int new (Queue *queue)
{
    queue->front = -1;
    queue->rear = -1;
}

int enqueue(Queue *queue, int val)
{
    if (queue->front > queue->rear)
    {
        printf("Queue Overflow");
    }
```

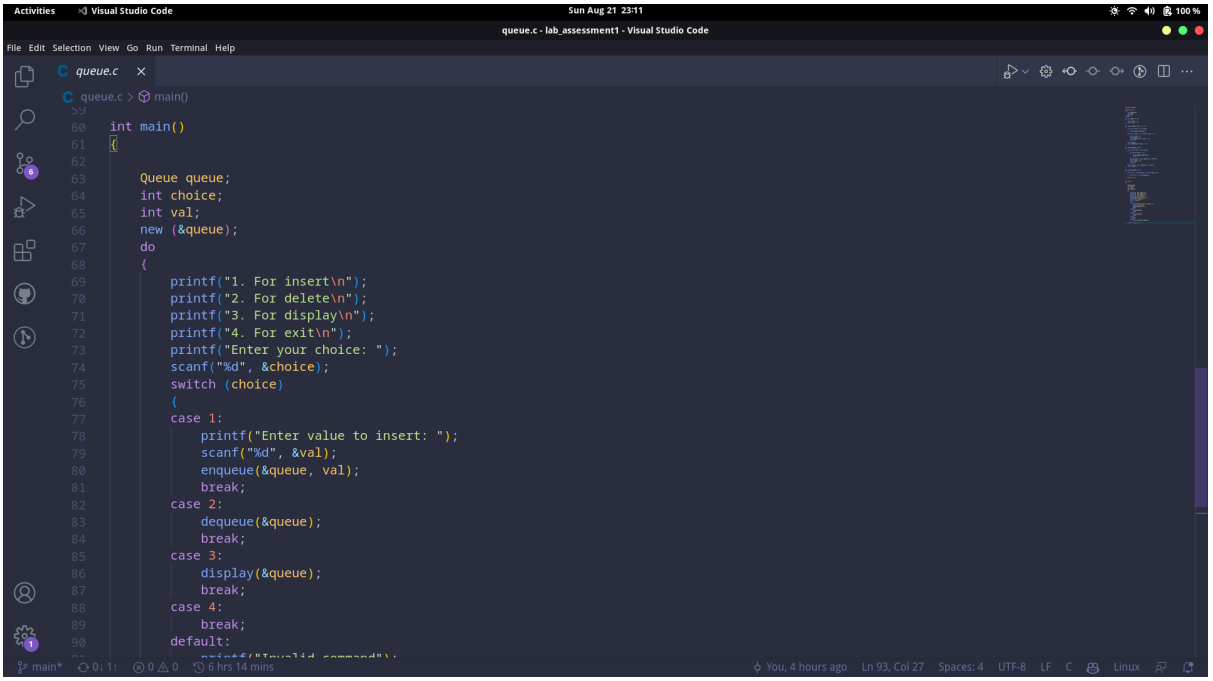
```
    if (queue->front == -1 && queue->rear == -1)
    {
        queue->front = 0;
        queue->rear = 0;
        queue->queue[queue->rear] = val;
        return 0;
    }
    queue->rear++;
    queue->queue[queue->rear] = val;
}

int dequeue(Queue *queue)
{
    if (queue->front == queue->rear)
    {
        if (queue->front == -1)
        {
            printf("Queue Underflow");
            return 0;
        }
        printf("%d\n", queue->queue[queue->front]);
        queue->front = -1;
        queue->rear = -1;
        return 0;
    }
    printf("%d\n", queue->queue[queue->front]);
    queue->front++;
}

int display(Queue *queue)
{
    for (int i = queue->front; i <= queue->rear; i++)
    {
        printf("%d ", queue->queue[i]);
    }
    printf("\n");
}

int main()
{

    Queue queue;
    int choice;
    int val;
    new (&queue);
    do
    {
        printf("1. For insert\n");
        printf("2. For delete\n");
        printf("3. For display\n");
        printf("4. For exit\n");
```

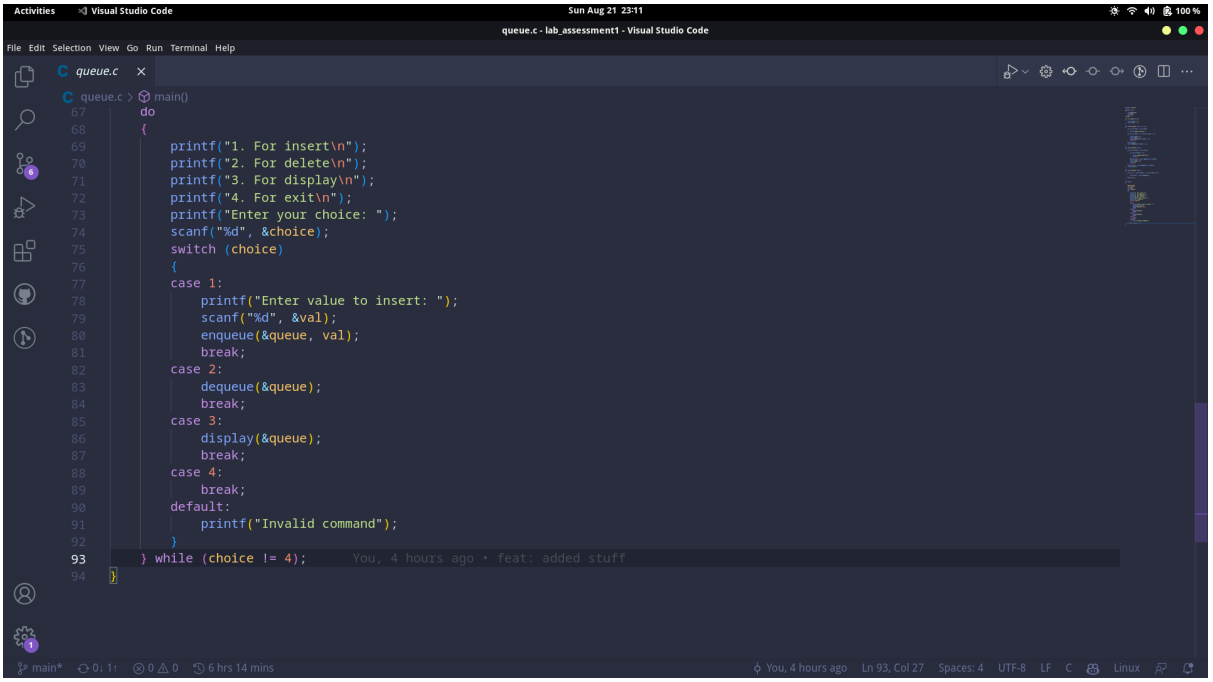
```
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            printf("Enter value to insert: ");
            scanf("%d", &val);
            enqueue(&queue, val);
            break;
        case 2:
            dequeue(&queue);
            break;
        case 3:
            display(&queue);
            break;
        case 4:
            break;
        default:
            printf("Invalid command");
        }
    } while (choice != 4);
}
```

Program Code:




Activities
Visual Studio Code
Sun Aug 21 23:11
queue.c - lab_assessment1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
queue.c
queue.c > main()
}
int dequeue(Queue *queue)
{
if (queue->front == queue->rear)
{
if (queue->front == -1)
{
printf("Queue Underflow");
return 0;
}
printf("%d\n", queue->queue[queue->front]);
queue->front = -1;
queue->rear = -1;
return 0;
}
printf("%d\n", queue->queue[queue->front]);
queue->front++;
}
int display(Queue *queue)
{
for (int i = queue->front; i <= queue->rear; i++)
{
printf("%d ", queue->queue[i]);
}
printf("\n");
}
int main()
{
main*
0↓ 1↑
0 0
6 hrs 14 mins
You, 4 hours ago
Ln 93, Col 27
Spaces: 4
UTF-8
LF
C
Linux

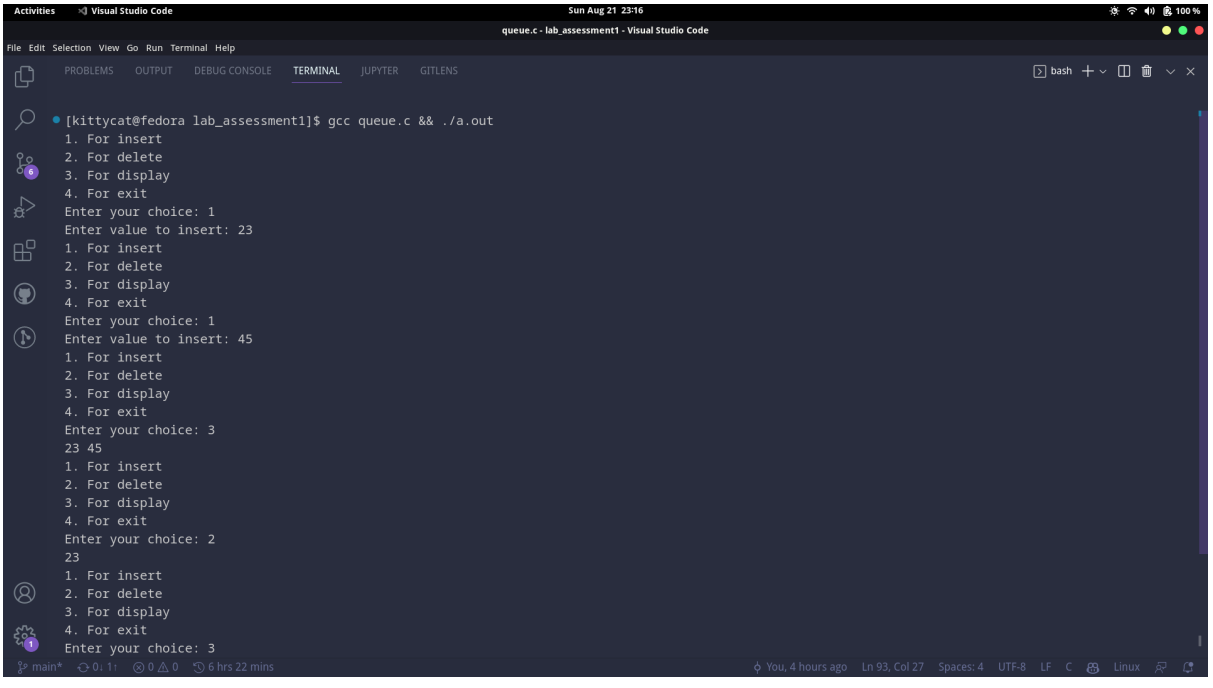

Activities
Visual Studio Code
Sun Aug 21 23:11
queue.c - lab_assessment1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
queue.c
queue.c > main()
int main()
{
Queue queue;
int choice;
int val;
new (&queue);
do
{
printf("1. For insert\n");
printf("2. For delete\n");
printf("3. For display\n");
printf("4. For exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);
switch (choice)
{
case 1:
printf("Enter value to insert: ");
scanf("%d", &val);
enqueue(&queue, val);
break;
case 2:
dequeue(&queue);
break;
case 3:
display(&queue);
break;
case 4:
break;
default:
main*
0↓ 1↑
0 0
6 hrs 14 mins
You, 4 hours ago
Ln 93, Col 27
Spaces: 4
UTF-8
LF
C
Linux

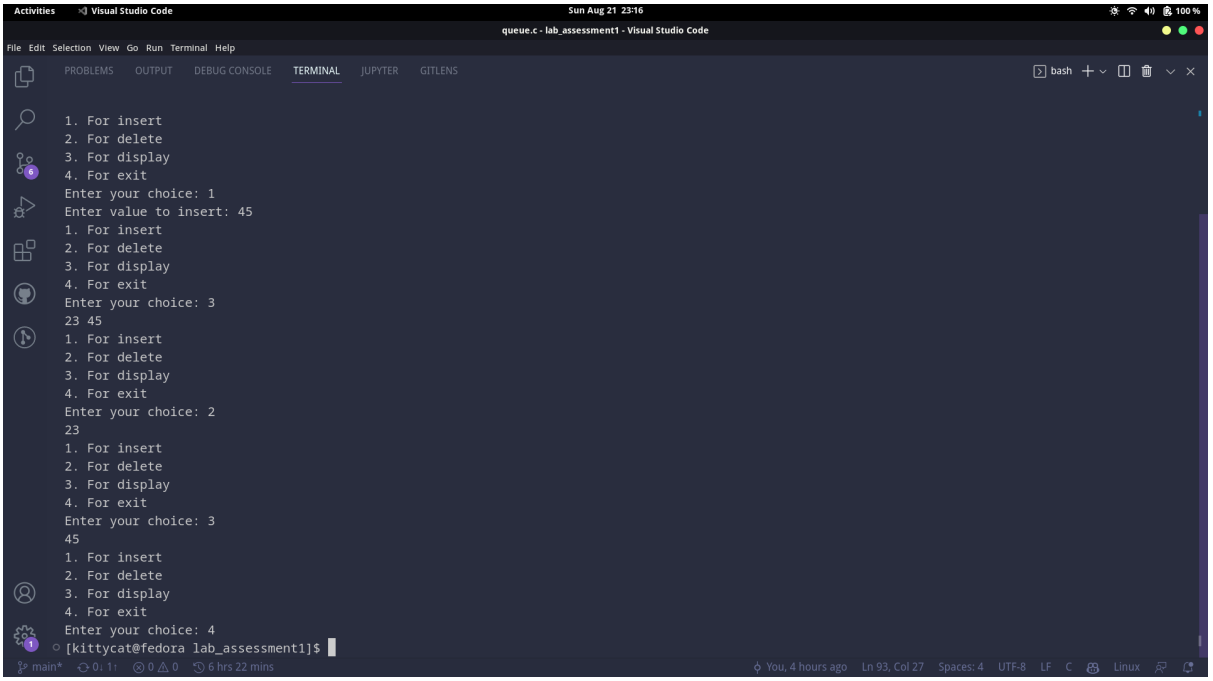

Activities
Visual Studio Code
Sun Aug 21 23:11
queue.c - lab_assessment1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
queue.c
queue.c > main()
do
{
printf("1. For insert\n");
printf("2. For delete\n");
printf("3. For display\n");
printf("4. For exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);
switch (choice)
{
case 1:
printf("Enter value to insert: ");
scanf("%d", &val);
enqueue(&queue, val);
break;
case 2:
dequeue(&queue);
break;
case 3:
display(&queue);
break;
case 4:
break;
default:
printf("Invalid command");
}
} while (choice != 4);
You, 4 hours ago • feat: added stuff
}
main*
0↓ 1↑
0 0
6 hrs 14 mins
You, 4 hours ago
Ln 93, Col 27
Spaces: 4
UTF-8
LF
C
Linux

Output:

```
[kittycat@fedora lab_assessment1]$ gcc queue.c && ./a.out
1. For insert
2. For delete
3. For display
4. For exit
Enter your choice: 1
Enter value to insert: 23
1. For insert
2. For delete
3. For display
4. For exit
Enter your choice: 1
Enter value to insert: 45
1. For insert
2. For delete
3. For display
4. For exit
Enter your choice: 3
23 45
1. For insert
2. For delete
3. For display
4. For exit
Enter your choice: 2
23
1. For insert
2. For delete
3. For display
4. For exit
Enter your choice: 3
45
1. For insert
2. For delete
3. For display
4. For exit
Enter your choice: 4
[kittycat@fedora lab_assessment1]$
```

Output Screenshot:





3.Write a program using arrays to perform insertion and deletion from both the ends of a Deque and also display the contents of it based on the choice given by the user.

Program Code:

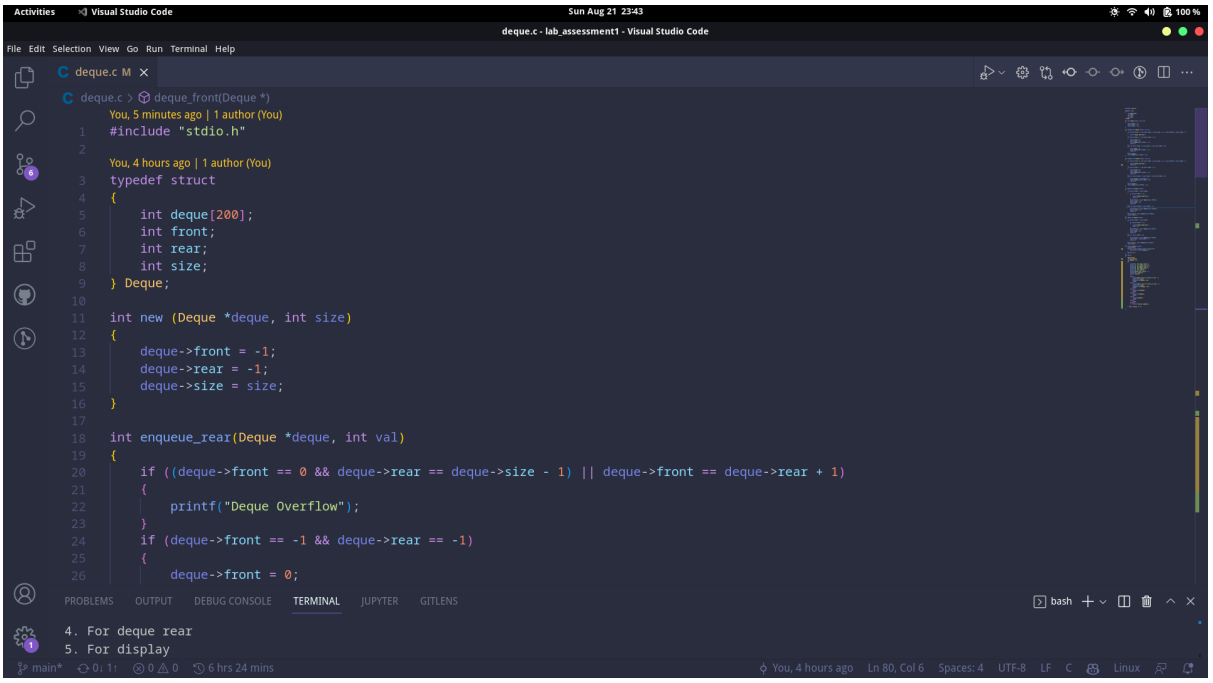
```
#include "stdio.h"

typedef struct
{
    int deque[200];
    int front;
    int rear;
    int size;
} Deque;

int new (Deque *deque, int size)
{
    deque->front = -1;
    deque->rear = -1;
    deque->size = size;
}

int enqueue_rear(Deque *deque, int val)
{
    if ((deque->front == 0 && deque->rear == deque->size - 1) ||
deque->front == deque->rear + 1)
    {
        printf("Deque Overflow");
    }
    if (deque->front == -1 && deque->rear == -1)
    {
        deque->front = 0;
        deque->rear = 0;
        deque->deque[deque->rear] = val;
        return 0;
    }
    else if (deque->rear == deque->size - 1 && deque->front != 0)
```

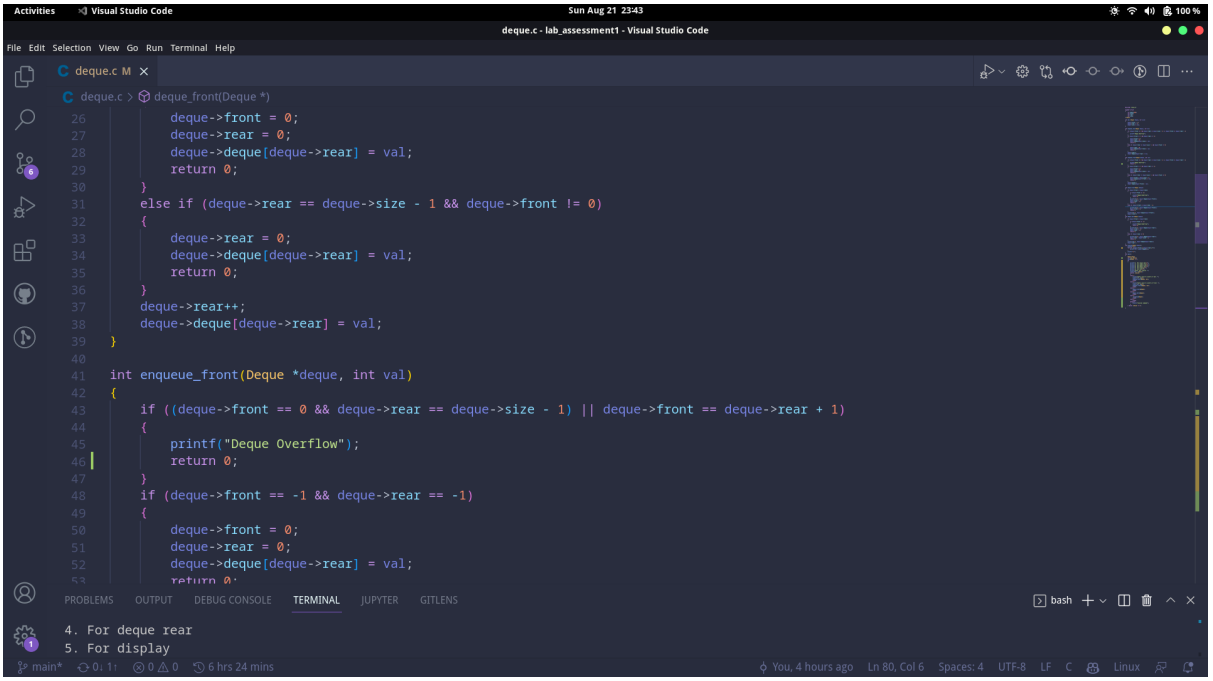
```
    {
        deque->rear = 0;
        deque->deque[deque->rear] = val;
        return 0;
    }
    deque->rear++;
    deque->deque[deque->rear] = val;
}

int enqueue_front(Deque *deque, int val)
{
    if ((deque->front == 0 && deque->rear == deque->size - 1) ||
deque->front == deque->rear + 1)
    {
        printf("Deque Overflow");
        return 0;
    }
    if (deque->front == -1 && deque->rear == -1)
    {
        deque->front = 0;
        deque->rear = 0;
        deque->deque[deque->rear] = val;
        return 0;
    }
    else if (deque->rear != deque->size - 1 && deque->front == 0)
    {
        deque->front = deque->size - 1;
        deque->deque[deque->front] = val;
        return 0;
    }
    deque->front--;
    deque->deque[deque->front] = val;
}

int deque_front(Deque *deque)
{
    if (deque->front == deque->rear)
    {
        if (deque->front == -1)
        {
            printf("Deque Underflow");
            return 0;
        }
        printf("%d\n", deque->deque[deque->front]);
        deque->front = -1;
        deque->rear = -1;
        return 0;
    }
    else if (deque->front == deque->rear - 1)
    {
        printf("%d\n", deque->deque[deque->front]);
```

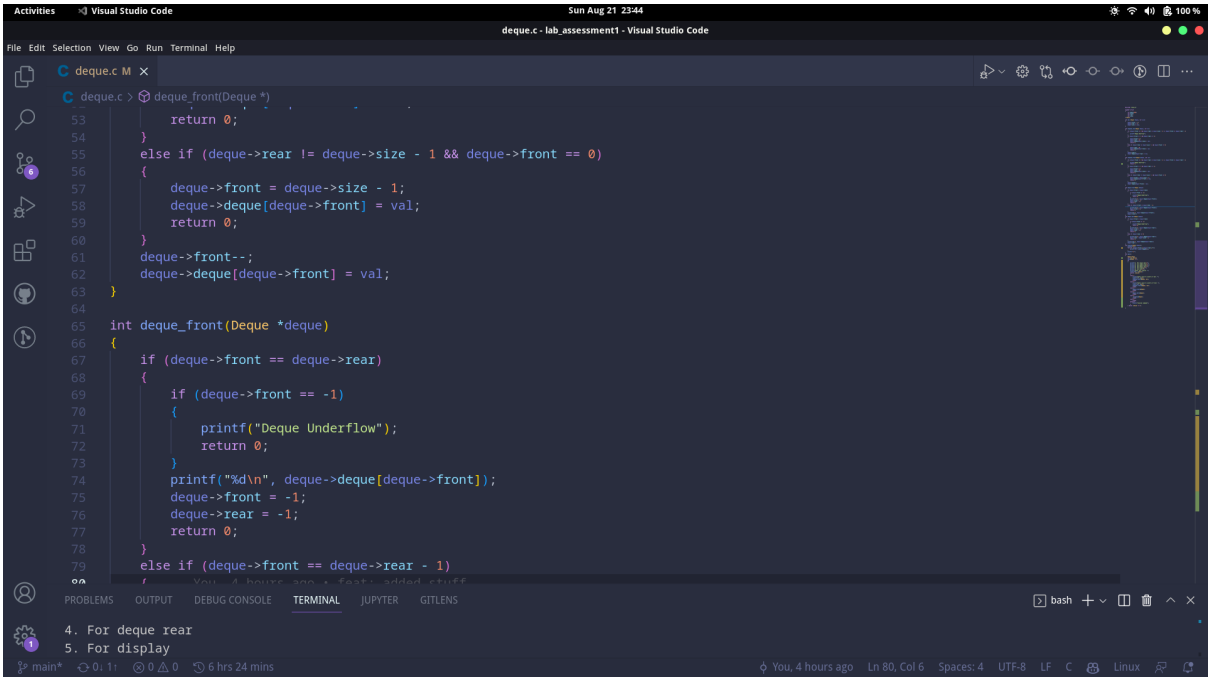
```
        deque->front = 0;
        return 0;
    }
    printf("%d\n", deque->deque[deque->front]);
    deque->front++;
}
int deque_rear(Deque *deque)
{
    if (deque->front == deque->rear)
    {
        if (deque->front == -1)
        {
            printf("Deque Underflow");
            return 0;
        }
        printf("%d\n", deque->deque[deque->rear]);
        deque->front = -1;
        deque->rear = -1;
        return 0;
    }
    else if (deque->rear == 0)
    {
        printf("%d\n", deque->deque[deque->rear]);
        deque->rear = deque->size - 1;
        return 0;
    }
    printf("%d\n", deque->deque[deque->rear]);
    deque->rear--;
}
int display(Deque *deque){
    printf("\n");
    for(int i=deque->front;i<=deque->rear;i++){
        printf("%d ",deque->deque[i]);
    }
    printf("\n");
}
int main()
{
    Deque deque;
    int val,choice;
    new(&deque, 37);
    do
    {
        printf("1. For enque rear\n");
        printf("2. For enque front\n");
        printf("3. For deque front\n");
        printf("4. For deque rear\n");
        printf("5. For display\n");
        printf("6. For exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
```

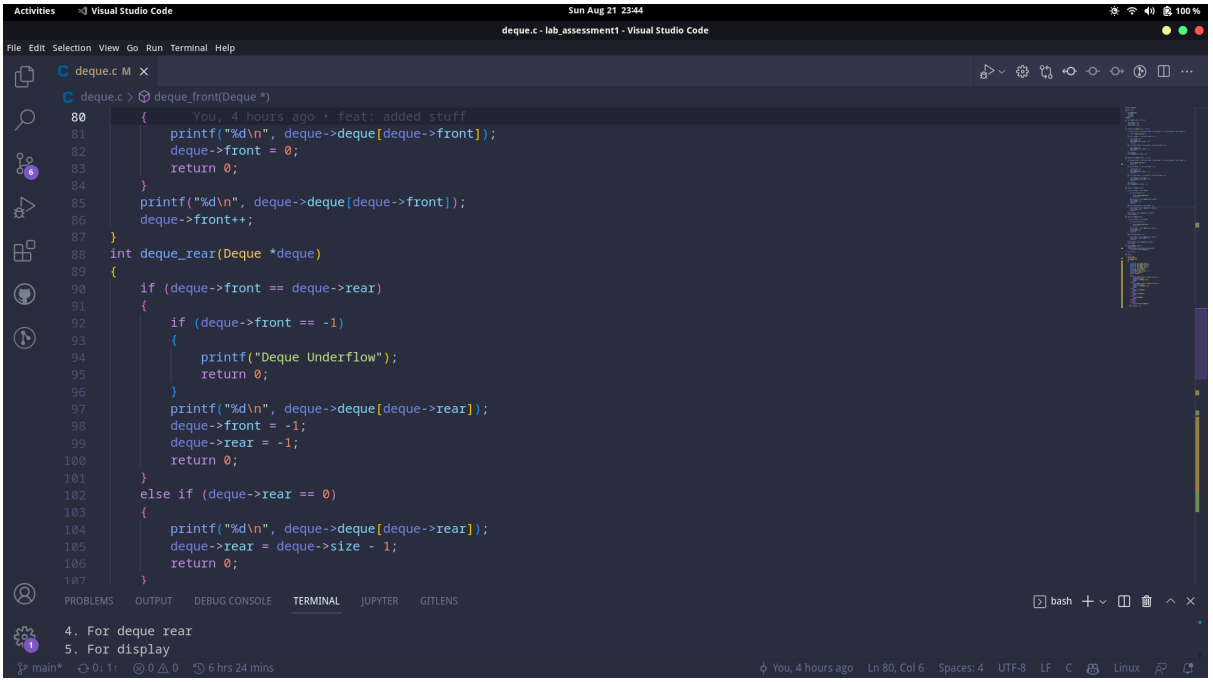
```
        switch (choice)
        {
        case 1:
            printf("Enter value to insert at rear: ");
            scanf("%d", &val);
            enqueue_rear(&deque, val);
            break;
        case 2:
            printf("Enter value to insert at front: ");
            scanf("%d", &val);
            enqueue_front(&deque, val);
            break;
        case 3:
            deque_front(&deque);
            break;
        case 4:
            deque_rear(&deque);
            break;
        case 5:
             display(&deque);
            break;
        case 6:
            break;
        default:
            printf("Invalid command");
        }
    } while (choice != 4);

}
```

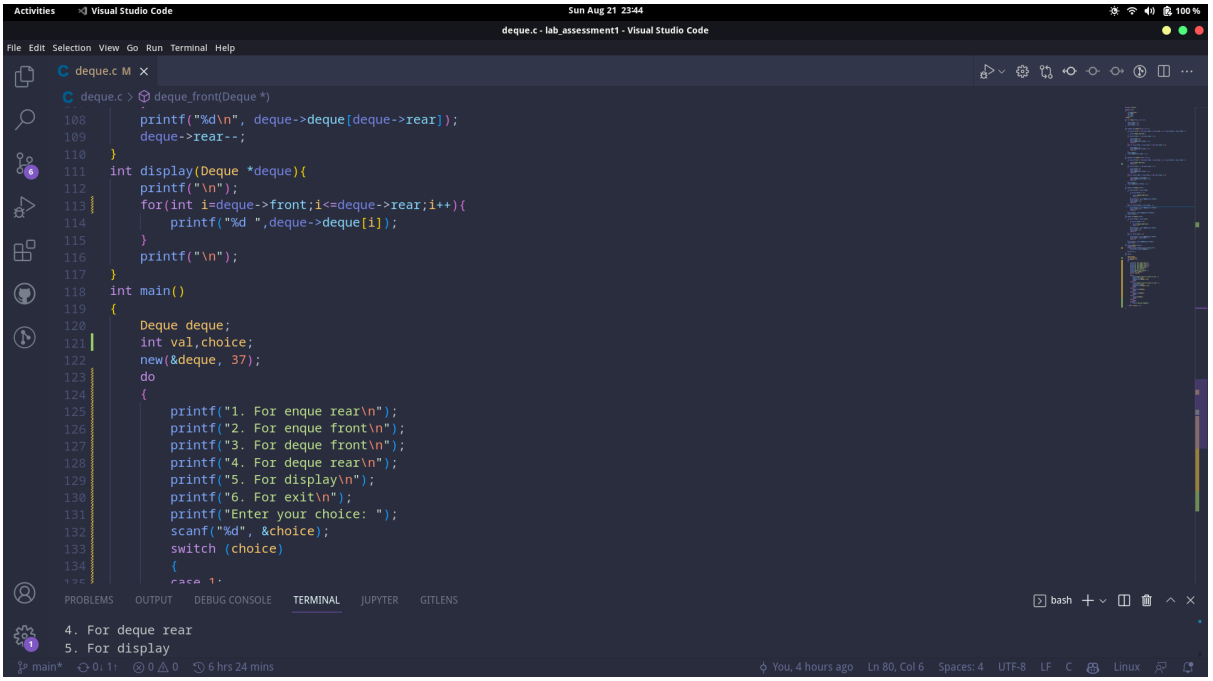
Program Screenshot:

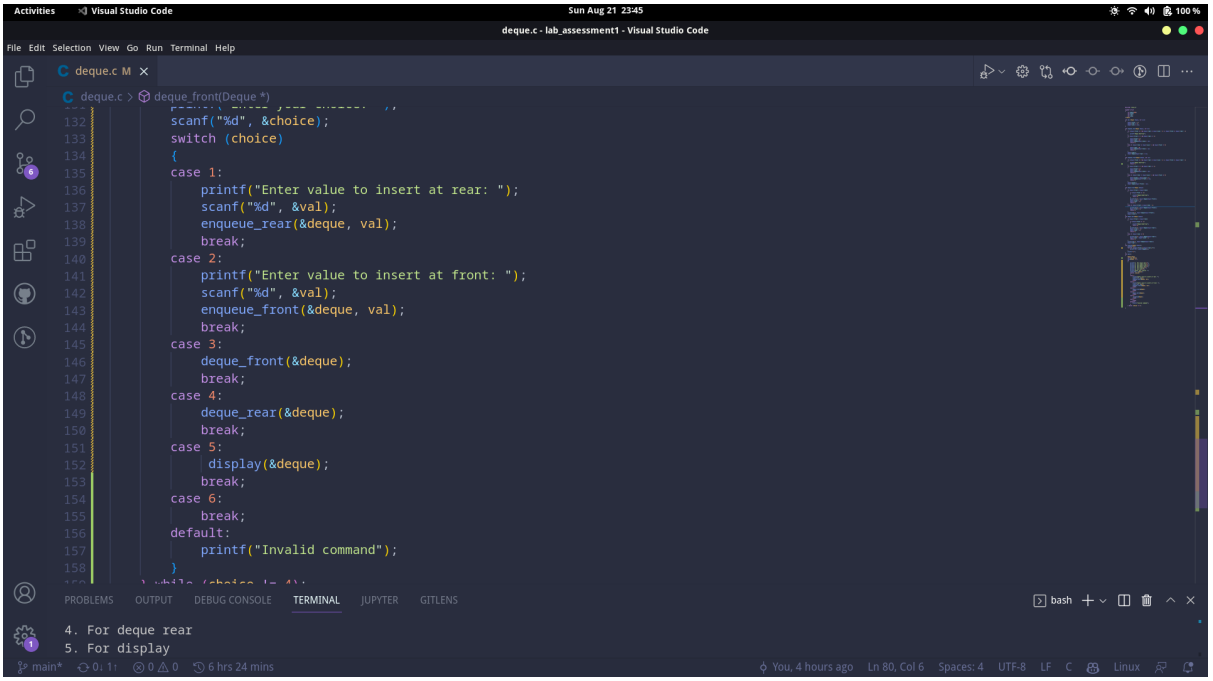


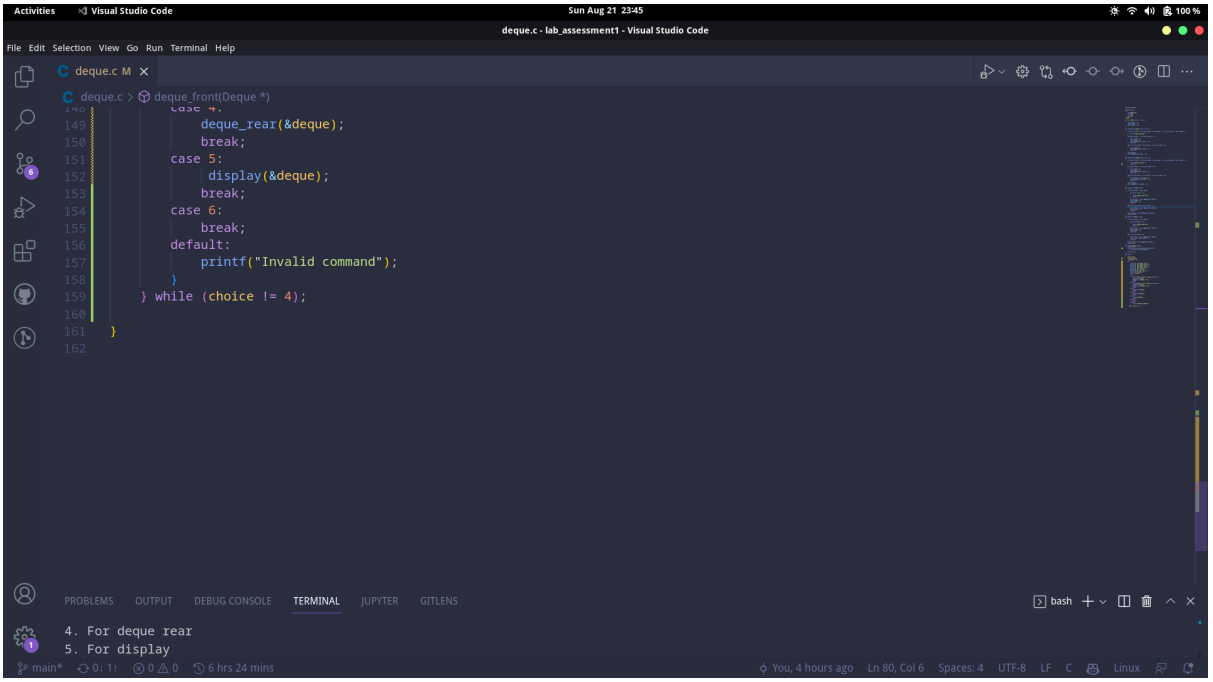

Activities
Visual Studio Code
Sun Aug 21 23:43
deque.c - lab_assessment1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
deque.c M
deque.c > deque_front(Deque *)
deque->front = 0;
deque->rear = 0;
deque->deque[deque->rear] = val;
return 0;
}
else if (deque->rear == deque->size - 1 && deque->front != 0)
{
deque->rear = 0;
deque->deque[deque->rear] = val;
return 0;
}
deque->rear++;
deque->deque[deque->rear] = val;
}
int enqueue_front(Deque *deque, int val)
{
if ((deque->front == 0 && deque->rear == deque->size - 1) || deque->front == deque->rear + 1)
{
printf("Deque Overflow");
return 0;
}
if (deque->front == -1 && deque->rear == -1)
{
deque->front = 0;
deque->rear = 0;
deque->deque[deque->rear] = val;
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER GITLENS
bash
4. For deque rear
5. For display
main*
0↓ 1↑
0 0
6 hrs 24 mins
You, 4 hours ago
Ln 80, Col 6
Spaces: 4
UTF-8
LF
C
Linux


Activities
Visual Studio Code
Sun Aug 21 23:44
deque.c - lab_assessment1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
deque.c M
deque.c > deque_front(Deque *)
return 0;
}
else if (deque->rear != deque->size - 1 && deque->front == 0)
{
deque->front = deque->size - 1;
deque->deque[deque->front] = val;
return 0;
}
deque->front--;
deque->deque[deque->front] = val;
}
int deque_front(Deque *deque)
{
if (deque->front == deque->rear)
{
if (deque->front == -1)
{
printf("Deque Underflow");
return 0;
}
printf("%d\n", deque->deque[deque->front]);
deque->front = -1;
deque->rear = -1;
return 0;
}
else if (deque->front == deque->rear - 1)
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER GITLENS
bash
4. For deque rear
5. For display
main*
0↓ 1↑
0 0
6 hrs 24 mins
You, 4 hours ago
Ln 80, Col 6
Spaces: 4
UTF-8
LF
C
Linux


Activities
Visual Studio Code
Sun Aug 21 23:44
deque.c - lab_assessment1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
deque.c M
deque.c > deque_front(Deque *)
{
You, 4 hours ago • feat: added stuff
printf("%d\n", deque->deque[deque->front]);
deque->front = 0;
return 0;
}
printf("%d\n", deque->deque[deque->front]);
deque->front++;
}
int deque_rear(Deque *deque)
{
if (deque->front == deque->rear)
{
if (deque->front == -1)
{
printf("Deque Underflow");
return 0;
}
printf("%d\n", deque->deque[deque->rear]);
deque->front = -1;
deque->rear = -1;
return 0;
}
else if (deque->rear == 0)
{
printf("%d\n", deque->deque[deque->rear]);
deque->rear = deque->size - 1;
return 0;
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER GITLENS
bash
4. For deque rear
5. For display
main*
0↓ 1↑
0 0
6 hrs 24 mins
You, 4 hours ago
Ln 80, Col 6
Spaces: 4
UTF-8
LF
C
Linux


Activities
Visual Studio Code
Sun Aug 21 23:44
deque.c - lab_assessment1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
deque.c M
deque.c > deque_front(Deque *)
printf("%d\n", deque->deque[deque->rear]);
deque->rear--;
}
int display(Deque *deque){
printf("\n");
for(int i=deque->front;i<=deque->rear;i++){
printf("%d ",deque->deque[i]);
}
printf("\n");
}
int main()
{
Deque deque;
int val,choice;
new(&deque, 37);
do
{
printf("1. For enque rear\n");
printf("2. For enque front\n");
printf("3. For deque front\n");
printf("4. For deque rear\n");
printf("5. For display\n");
printf("6. For exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);
switch (choice)
{
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER GITLENS
bash
4. For deque rear
5. For display
main* 0↓ 1↑ 0 0 6 hrs 24 mins
You, 4 hours ago Ln 80, Col 6 Spaces: 4 UTF-8 LF C Linux


Activities
Visual Studio Code
Sun Aug 21 23:45
deque.c - lab_assessment1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
deque.c M
deque.c > deque_front(Deque *)
scanf("%d", &choice);
switch (choice)
{
case 1:
printf("Enter value to insert at rear: ");
scanf("%d", &val);
enqueue_rear(&deque, val);
break;
case 2:
printf("Enter value to insert at front: ");
scanf("%d", &val);
enqueue_front(&deque, val);
break;
case 3:
deque_front(&deque);
break;
case 4:
deque_rear(&deque);
break;
case 5:
display(&deque);
break;
case 6:
break;
default:
printf("Invalid command");
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER GITLENS
bash
4. For deque rear
5. For display
main* 0↓ 1↑ 0 0 6 hrs 24 mins
You, 4 hours ago Ln 80, Col 6 Spaces: 4 UTF-8 LF C Linux


Activities
Visual Studio Code
Sun Aug 21 23:45
deque.c - lab_assessment1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
deque.c M
deque.c > deque_front(Deque *)
deque_rear(&deque);
break;
case 5:
display(&deque);
break;
case 6:
break;
default:
printf("Invalid command");
}
} while (choice != 4);
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER GITLENS
bash
4. For deque rear
5. For display
main* 0↓ 1↑ 0 0 6 hrs 24 mins
You, 4 hours ago Ln 80, Col 6 Spaces: 4 UTF-8 LF C Linux

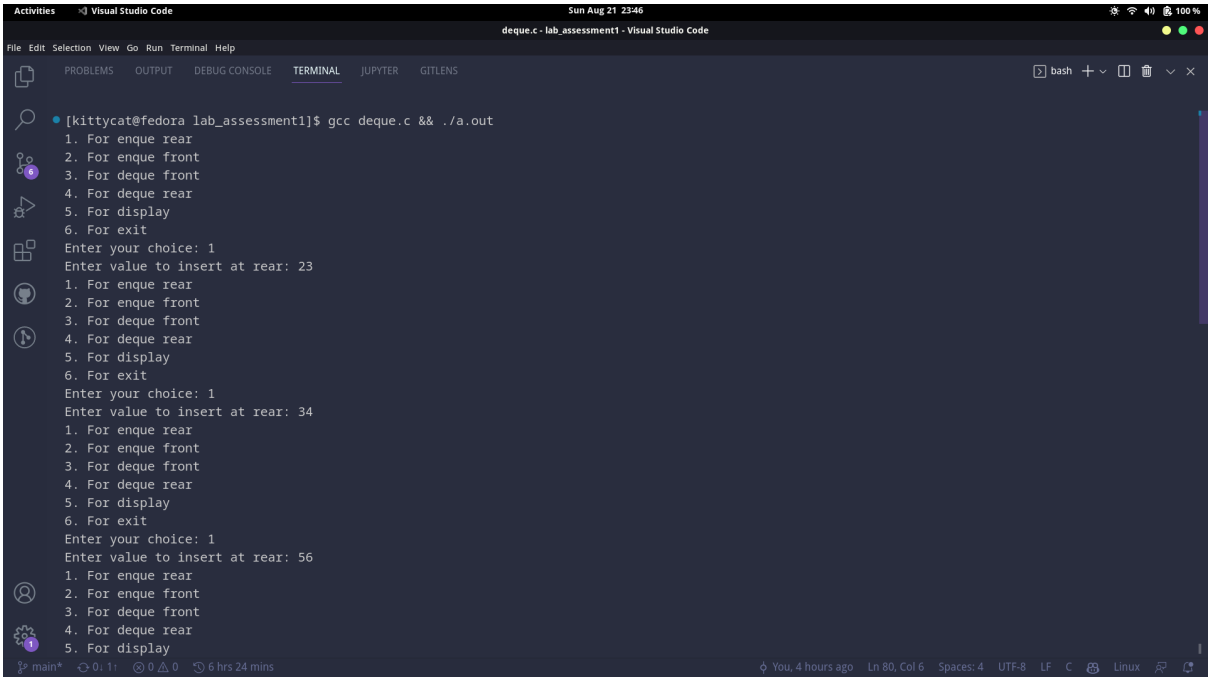
Output:

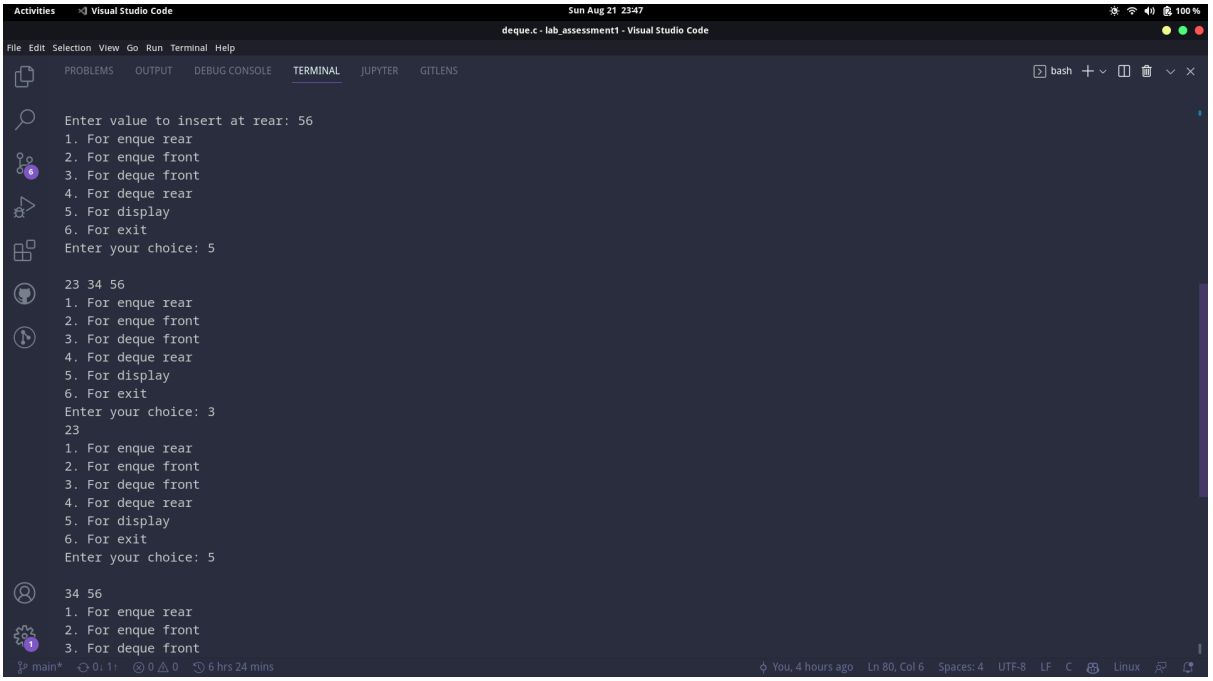
```
[kittycat@fedora lab_assessment1]$ gcc deque.c && ./a.out
1. For enque rear
2. For enque front
3. For deque front
4. For deque rear
5. For display
6. For exit
Enter your choice: 1
Enter value to insert at rear: 23
1. For enque rear
2. For enque front
3. For deque front
4. For deque rear
5. For display
6. For exit
Enter your choice: 1
Enter value to insert at rear: 34
1. For enque rear
2. For enque front
3. For deque front
4. For deque rear
5. For display
6. For exit
Enter your choice: 1
Enter value to insert at rear: 56
1. For enque rear
2. For enque front
3. For deque front
4. For deque rear
5. For display
6. For exit
Enter your choice: 5

23 34 56
1. For enque rear
2. For enque front
3. For deque front
4. For deque rear
5. For display
6. For exit
Enter your choice: 3
23
1. For enque rear
2. For enque front
3. For deque front
4. For deque rear
5. For display
6. For exit
Enter your choice: 5

34 56
1. For enque rear
2. For enque front
3. For deque front
4. For deque rear
5. For display
```

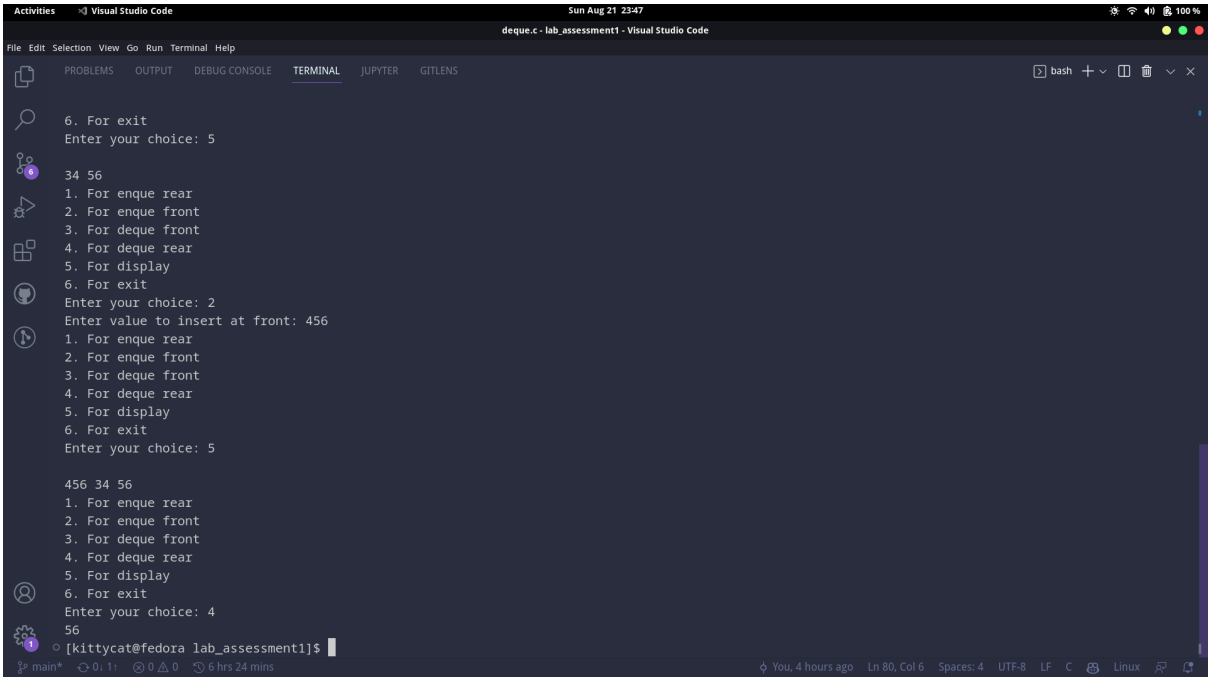
6. For exit
Enter your choice: 2
Enter value to insert at front: 456
1. For enque rear
2. For enque front
3. For deque front
4. For deque rear
5. For display
6. For exit
Enter your choice: 5

456 34 56
1. For enque rear
2. For enque front
3. For deque front
4. For deque rear
5. For display
6. For exit
Enter your choice: 4
56

Output Screenshot:







4.Write a program that takes the details of mobile phone(model name,year,camera resolution,RAM , memory card size and Operating system)and sort the mobile phones in ascending order based on their RAM size using insertion sort.

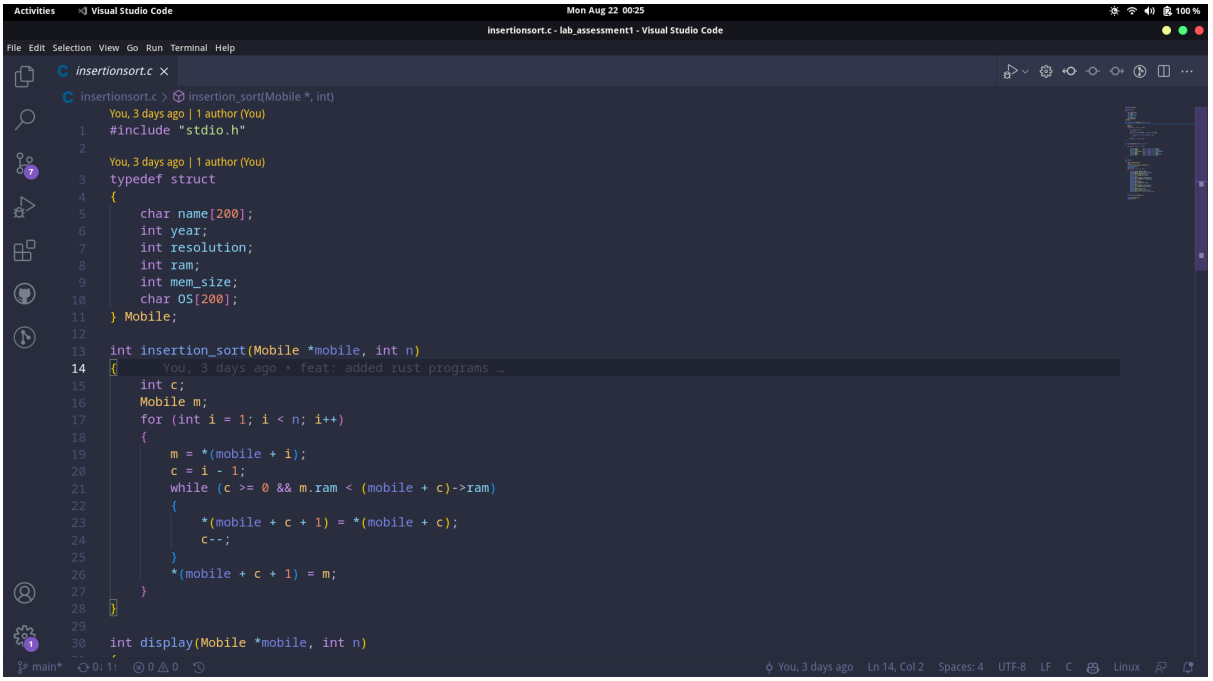
Program Code:

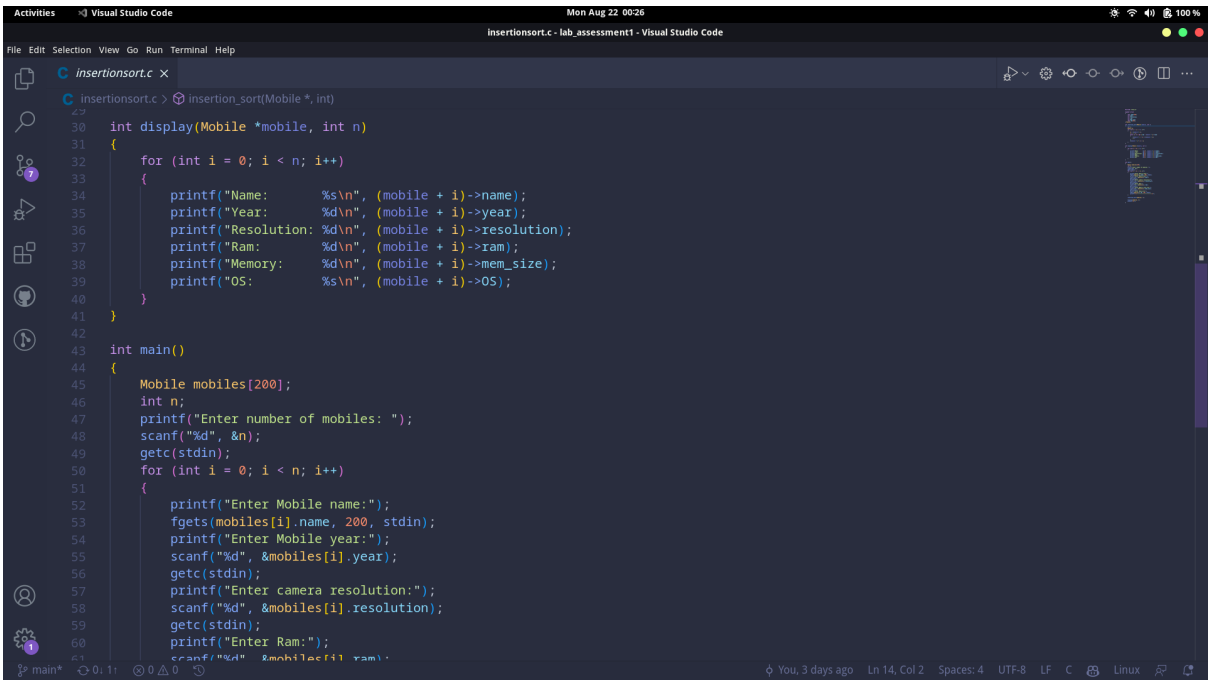
```
#include "stdio.h"

typedef struct
{
    char name[200];
    int year;
    int resolution;
    int ram;
    int mem_size;
    char OS[200];
} Mobile;

int insertion_sort(Mobile *mobile, int n)
{
    int c;
```

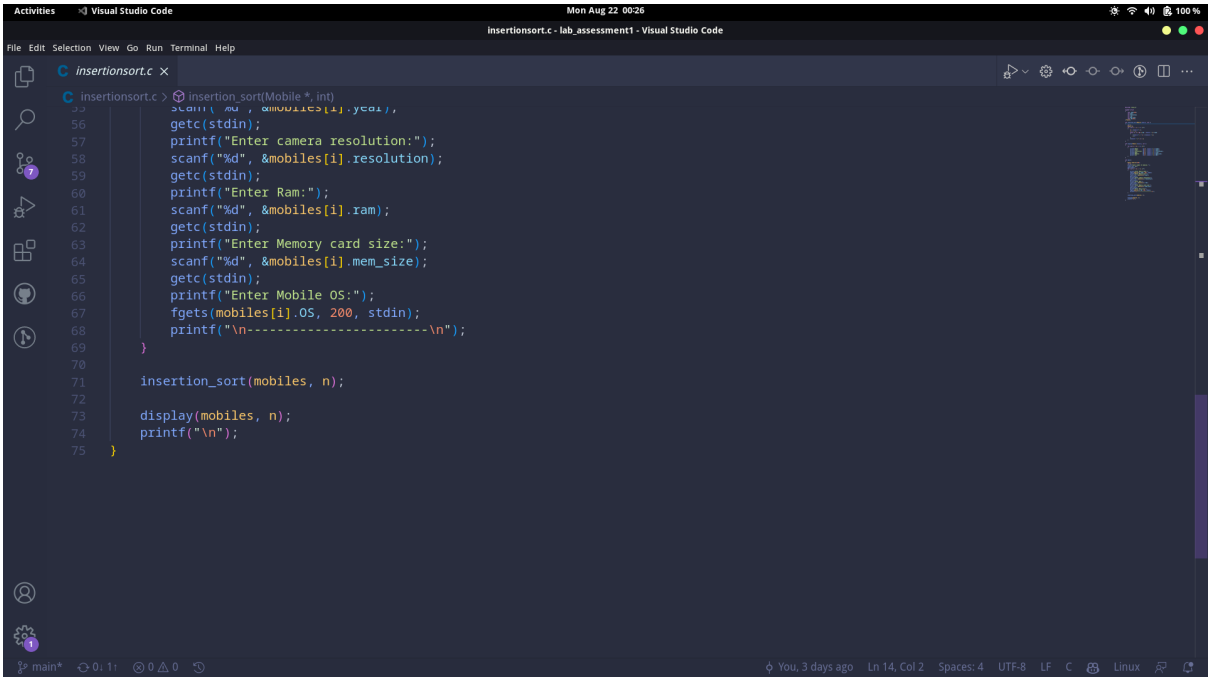
```
    Mobile m;
    for (int i = 1; i < n; i++)
    {
        m = *(mobile + i);
        c = i - 1;
        while (c >= 0 && m.ram < (mobile + c)->ram)
        {
            *(mobile + c + 1) = *(mobile + c);
            c--;
        }
        *(mobile + c + 1) = m;
    }
}

int display(Mobile *mobile, int n)
{
    for (int i = 0; i < n; i++)
    {
        printf("Name:       %s\n", (mobile + i)->name);
        printf("Year:       %d\n", (mobile + i)->year);
        printf("Resolution: %d\n", (mobile + i)->resolution);
        printf("Ram:        %d\n", (mobile + i)->ram);
        printf("Memory:     %d\n", (mobile + i)->mem_size);
        printf("OS:         %s\n", (mobile + i)->OS);
    }
}

int main()
{
    Mobile mobiles[200];
    int n;
    printf("Enter number of mobiles: ");
    scanf("%d", &n);
    getc(stdin);
    for (int i = 0; i < n; i++)
    {
        printf("Enter Mobile name:");
        fgets(mobiles[i].name, 200, stdin);
        printf("Enter Mobile year:");
        scanf("%d", &mobiles[i].year);
        getc(stdin);
        printf("Enter camera resolution:");
        scanf("%d", &mobiles[i].resolution);
        getc(stdin);
        printf("Enter Ram:");
        scanf("%d", &mobiles[i].ram);
        getc(stdin);
        printf("Enter Memory card size:");
        scanf("%d", &mobiles[i].mem_size);
        getc(stdin);
        printf("Enter Mobile OS:");
```

```
        fgets(mobiles[i].OS, 200, stdin);
        printf("\n-------------------------\n");
    }

    insertion_sort(mobiles, n);

    display(mobiles, n);
    printf("\n");
}
```

Program Screenshot:







Output:

[kittycat@fedora lab_assessment1]$ gcc insertionsort.c && ./a.out
Enter number of mobiles: 4
Enter Mobile name:Samsung M40
Enter Mobile year:2021
Enter camera resolution:45
Enter Ram:8
Enter Memory card size:128
Enter Mobile OS:Android 10

Enter Mobile name:Apple 10
Enter Mobile year:2020
Enter camera resolution:34
Enter Ram:6
Enter Memory card size:64
Enter Mobile OS:IOS

Enter Mobile name:Samsung M20
Enter Mobile year:2019
Enter camera resolution:20
Enter Ram:4
Enter Memory card size:64
Enter Mobile OS:Android 9

Enter Mobile name:OPPO
Enter Mobile year:2019
Enter camera resolution:23
Enter Ram:2
Enter Memory card size:32
Enter Mobile OS:Android

Name: OPPO

Year: 2019

Resolution: 23
Ram: 2
Memory: 32
OS: Android

Name: Samsung M20

Year: 2019
Resolution: 20
Ram: 4
Memory: 64
OS: Android 9

Name: Apple 10

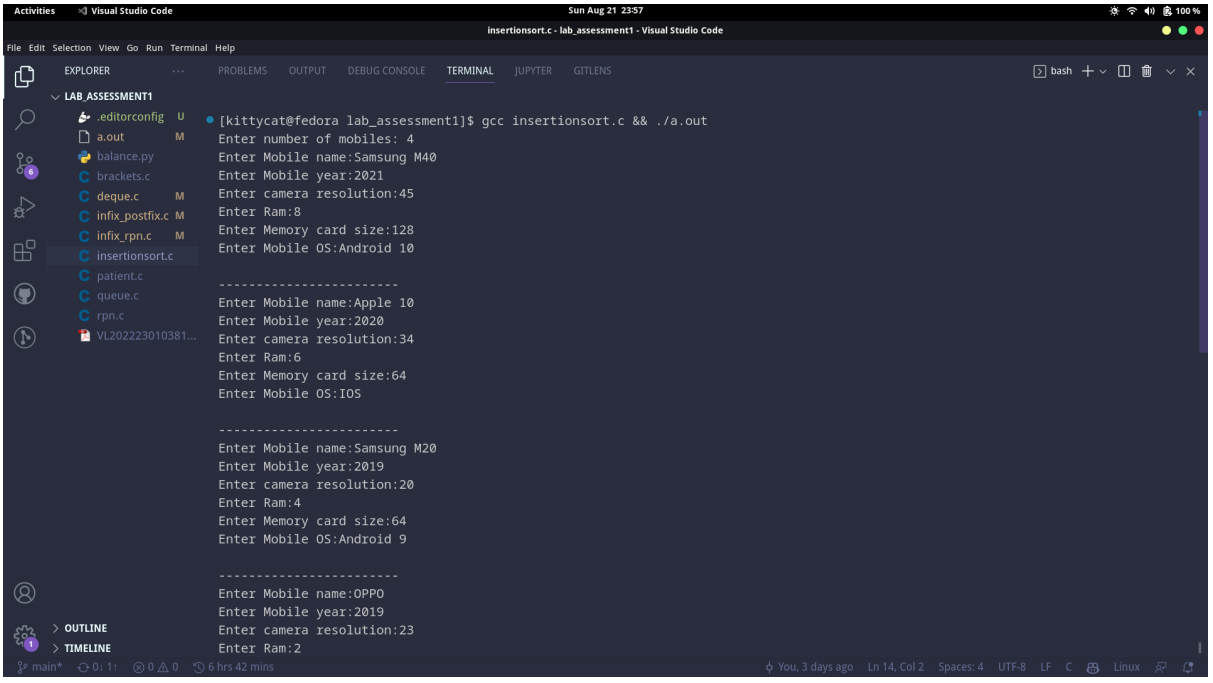
Year: 2020
Resolution: 34
Ram: 6
Memory: 64
OS: IOS

Name: Samsung M40

Year: 2021
Resolution: 45
Ram: 8
Memory: 128
OS: Android 10

[kittycat@fedora lab_assessment1]$

Output Screenshot:



5. Write a program that takes the details of a patient(hospital number, patient name, age, token number, height, , weight, reason(disease) and sort the patients in ascending order based on their token number using quick sort.

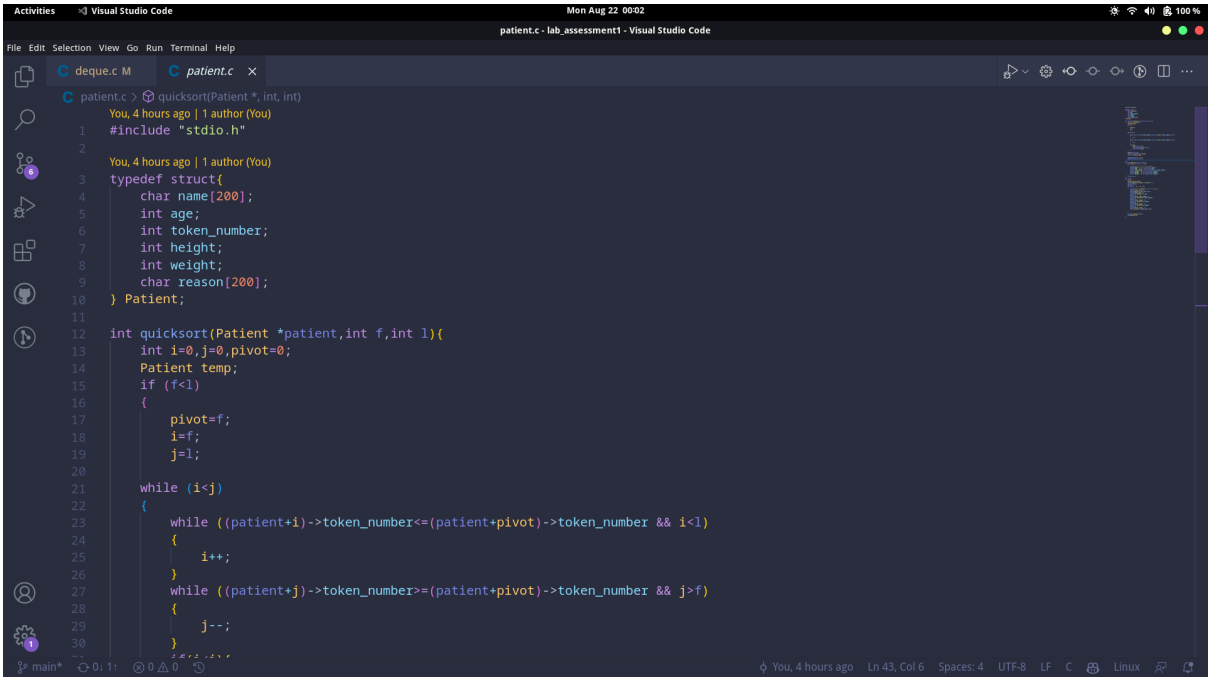
Program Code:

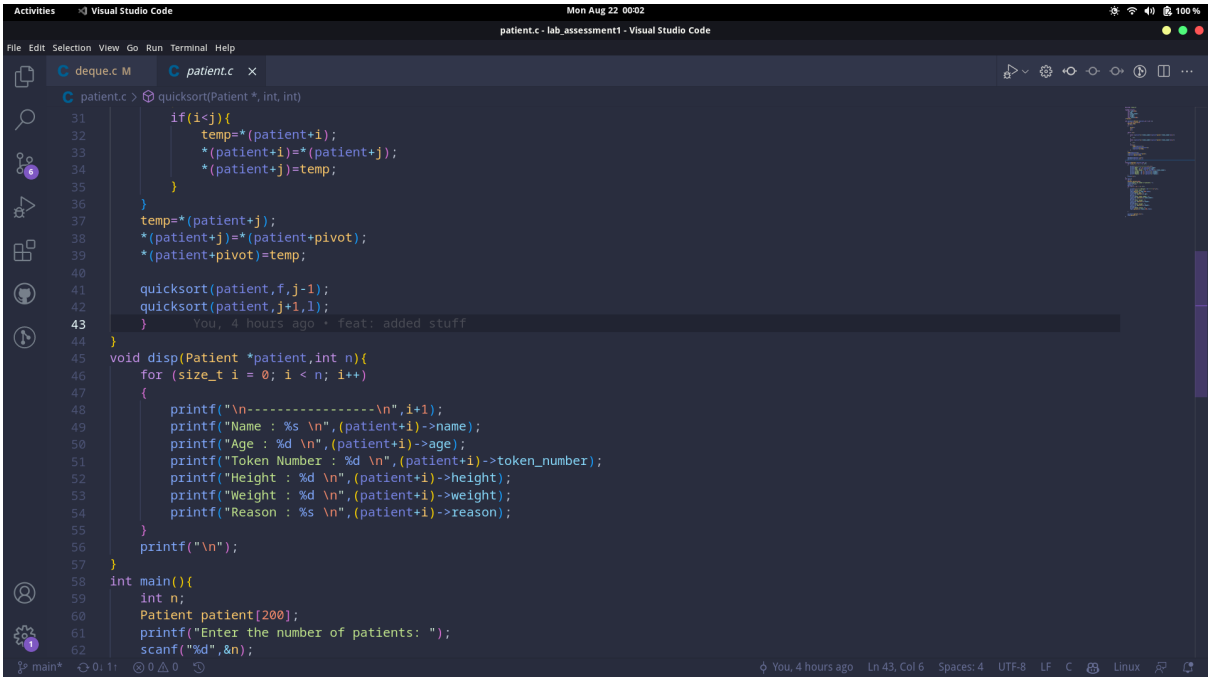
```
#include "stdio.h"

typedef struct{
    char name[200];
    int age;
    int token_number;
    int height;
    int weight;
    char reason[200];
} Patient;

int quicksort(Patient *patient,int f,int l){
    int i=0,j=0,pivot=0;
    Patient temp;
    if (f<l)
```

```
    {
        pivot=f;
        i=f;
        j=l;

    while (i<j)
    {
        while ((patient+i)->token_number<=(patient+pivot)->token_number
&& i<l)
        {
            i++;
        }
        while ((patient+j)->token_number>=(patient+pivot)->token_number
&& j>f)
        {
            j--;
        }
        if(i<j){
            temp=*(patient+i);
            *(patient+i)=*(patient+j);
            *(patient+j)=temp;
        }
    }
    temp=*(patient+j);
    *(patient+j)=*(patient+pivot);
    *(patient+pivot)=temp;

    quicksort(patient,f,j-1);
    quicksort(patient,j+1,l);
    }
}
void disp(Patient *patient,int n){
    for (size_t i = 0; i < n; i++)
    {
        printf("\n-----------------\n",i+1);
        printf("Name : %s \n",(patient+i)->name);
        printf("Age : %d \n",(patient+i)->age);
        printf("Token Number : %d \n",(patient+i)->token_number);
        printf("Height : %d \n",(patient+i)->height);
        printf("Weight : %d \n",(patient+i)->weight);
        printf("Reason : %s \n",(patient+i)->reason);
    }
    printf("\n");
}
int main(){
    int n;
    Patient patient[200];
    printf("Enter the number of patients: ");
    scanf("%d",&n);
    getchar();
    for (int i = 0; i < n; i++)
```

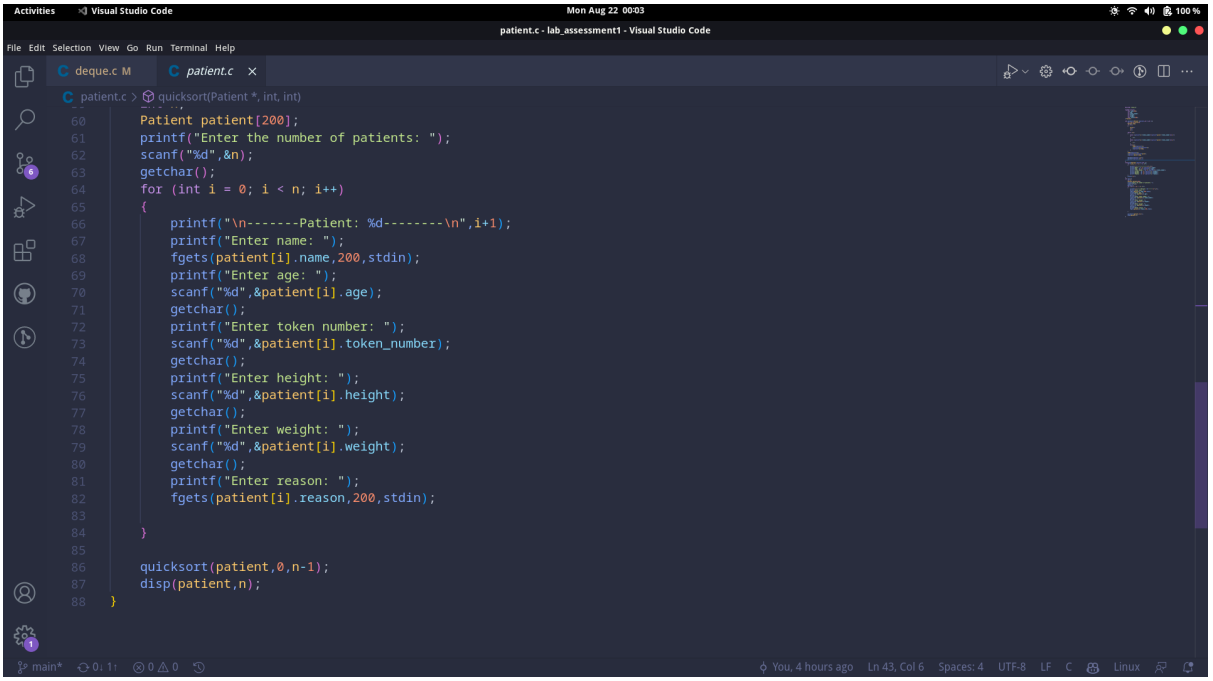
```
    {
        printf("\n-------Patient: %d--------\n",i+1);
        printf("Enter name: ");
        fgets(patient[i].name,200,stdin);
        printf("Enter age: ");
        scanf("%d",&patient[i].age);
        getchar();
        printf("Enter token number: ");
        scanf("%d",&patient[i].token_number);
        getchar();
        printf("Enter height: ");
        scanf("%d",&patient[i].height);
        getchar();
        printf("Enter weight: ");
        scanf("%d",&patient[i].weight);
        getchar();
        printf("Enter reason: ");
        fgets(patient[i].reason,200,stdin);

    }

    quicksort(patient,0,n-1);
    disp(patient,n);
}
```

Program Screenshot:





```
        if(i<j){
            temp=*(patient+i);
            *(patient+i)=*(patient+j);
            *(patient+j)=temp;
        }
    }
    temp=*(patient+j);
    *(patient+j)=*(patient+pivot);
    *(patient+pivot)=temp;

    quicksort(patient,f,j-1);
    quicksort(patient,j+1,l);
    }       You, 4 hours ago • feat: added stuff
}
void disp(Patient *patient,int n){
    for (size_t i = 0; i < n; i++)
    {
        printf("\n----------------\n",i+1);
        printf("Name : %s \n",(patient+i)->name);
        printf("Age : %d \n",(patient+i)->age);
        printf("Token Number : %d \n",(patient+i)->token_number);
        printf("Height : %d \n",(patient+i)->height);
        printf("Weight : %d \n",(patient+i)->weight);
        printf("Reason : %s \n",(patient+i)->reason);
    }
    printf("\n");
}
int main(){
    int n;
    Patient patient[200];
    printf("Enter the number of patients: ");
    scanf("%d",&n);
```



```
    Patient patient[200];
    printf("Enter the number of patients: ");
    scanf("%d",&n);
    getchar();
    for (int i = 0; i < n; i++)
    {
        printf("\n-------Patient: %d--------\n",i+1);
        printf("Enter name: ");
        fgets(patient[i].name,200,stdin);
        printf("Enter age: ");
        scanf("%d",&patient[i].age);
        getchar();
        printf("Enter token number: ");
        scanf("%d",&patient[i].token_number);
        getchar();
        printf("Enter height: ");
        scanf("%d",&patient[i].height);
        getchar();
        printf("Enter weight: ");
        scanf("%d",&patient[i].weight);
        getchar();
        printf("Enter reason: ");
        fgets(patient[i].reason,200,stdin);

    }

    quicksort(patient,0,n-1);
    disp(patient,n);
}
```

Program Output:

[kittycat@fedora lab_assessment1]$ gcc patient.c && ./a.out
Enter the number of patients: 4

-------Patient: 1--------
Enter name: Utkarsh
Enter age: 20
Enter token number: 45
Enter height: 180
Enter weight: 89
Enter reason: Obesity

-------Patient: 2--------
Enter name: Ujjawal
Enter age: 19
Enter token number: 25
Enter height: 175
Enter weight: 67
Enter reason: Depression

-------Patient: 3--------

Enter name: Tejas
Enter age: 18
Enter token number: 12
Enter height: 165
Enter weight: 56
Enter reason: Insomnia

-------Patient: 4--------
Enter name: Murali
Enter age: 45
Enter token number: 34
Enter height: 125
Enter weight: 67
Enter reason: Obesity

Name : Tejas

Age : 18
Token Number : 12
Height : 165
Weight : 56
Reason : Insomnia

Name : Ujjawal

Age : 19
Token Number : 25
Height : 175
Weight : 67
Reason : Depression

Name : Murali

Age : 45
Token Number : 34
Height : 125
Weight : 67
Reason : Obesity

Name : Utkarsh

Age : 20
Token Number : 45
Height : 180
Weight : 89
Reason : Obesity

[kittycat@fedora lab_assessment1]$

Output Screenshot: