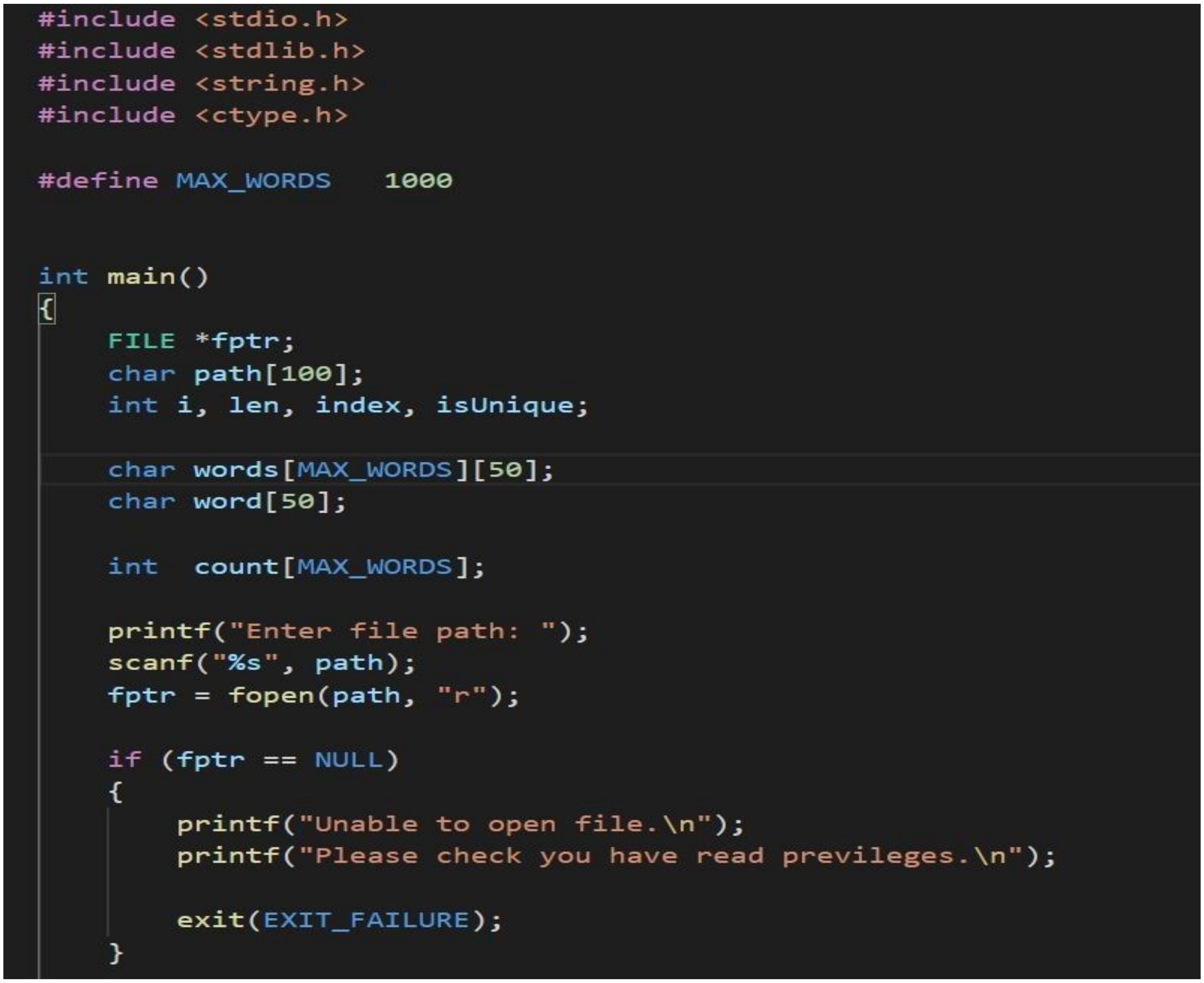
# OS External Practical Exam

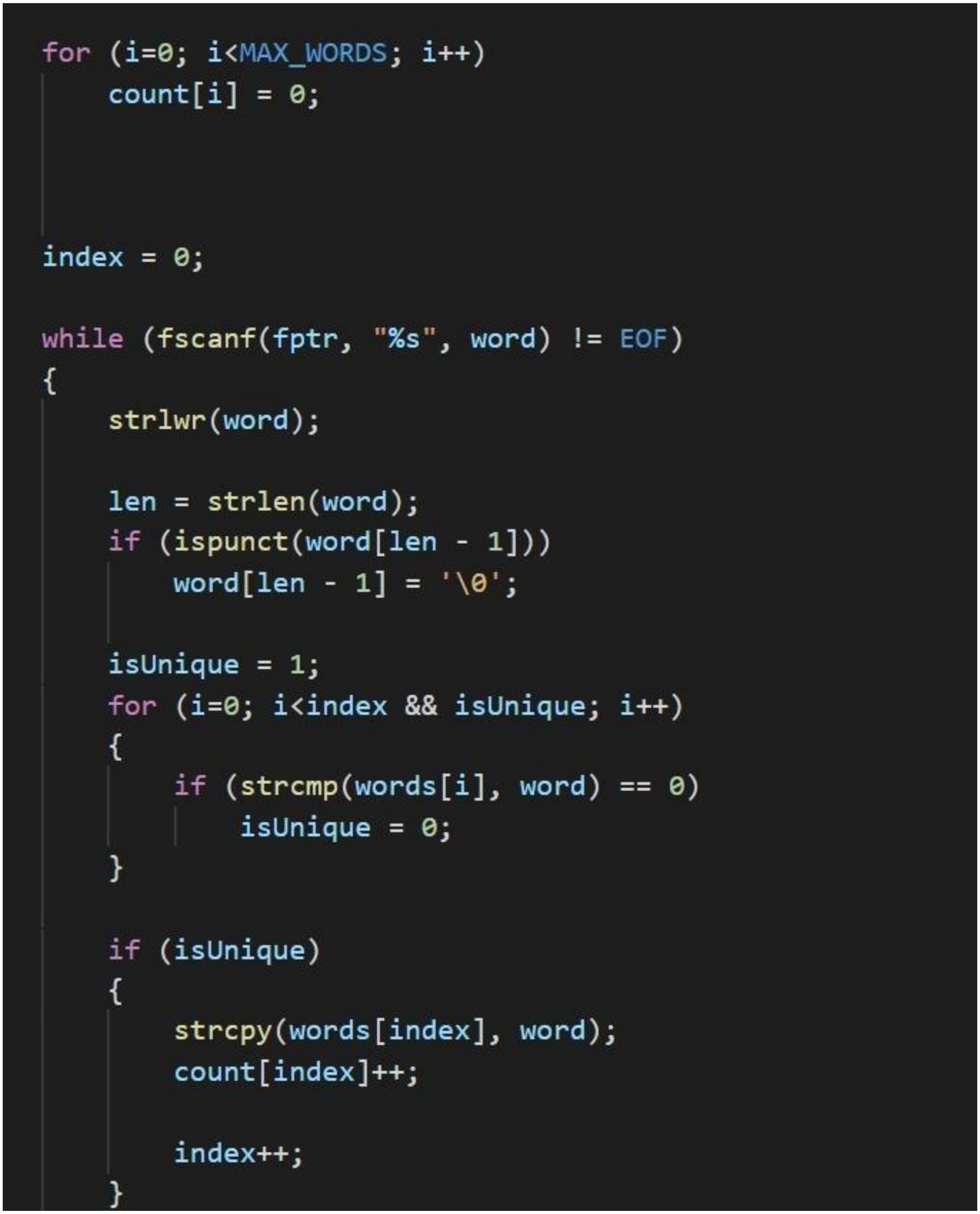
Name- Arpit Agarwal

Enrollment No-A20405220102

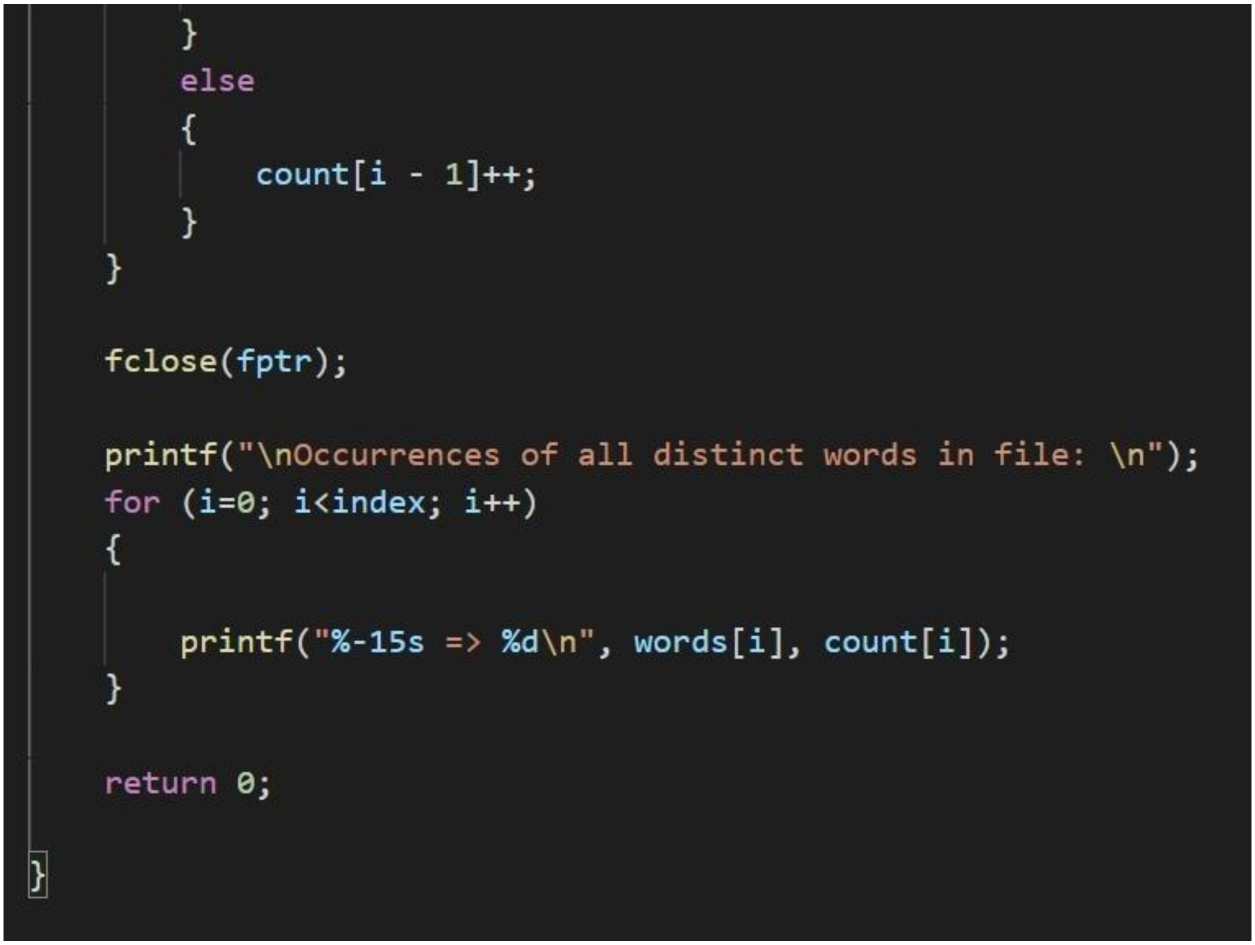
Q1. Write a C program to count the occurrence of each word and total words in a file.



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

#define MAX_WORDS   1000


int main()
{
    FILE *fptr;
    char path[100];
    int i, len, index, isUnique;

    char words[MAX_WORDS][50];
    char word[50];

    int  count[MAX_WORDS];

    printf("Enter file path: ");
    scanf("%s", path);
    fptr = fopen(path, "r");

    if (fptr == NULL)
    {
        printf("Unable to open file.\n");
        printf("Please check you have read previleges.\n");

        exit(EXIT_FAILURE);
    }
```

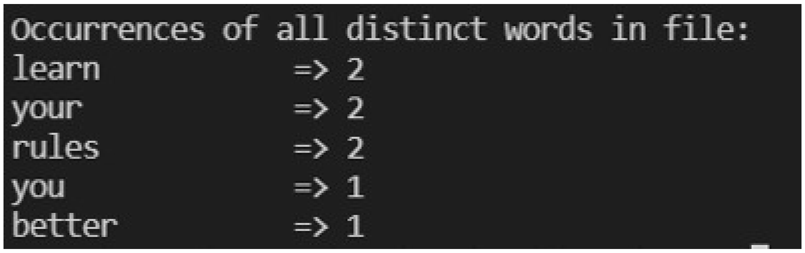


```
for (i=0; i<MAX_WORDS; i++)
    count[i] = 0;



index = 0;

while (fscanf(fptr, "%s", word) != EOF)
{
    strlwr(word);

    len = strlen(word);
    if (ispunct(word[len - 1]))
        word[len - 1] = '\0';

    isUnique = 1;
    for (i=0; i<index && isUnique; i++)
    {
        if (strcmp(words[i], word) == 0)
            isUnique = 0;
    }

    if (isUnique)
    {
        strcpy(words[index], word);
        count[index]++;

        index++;
    }
```



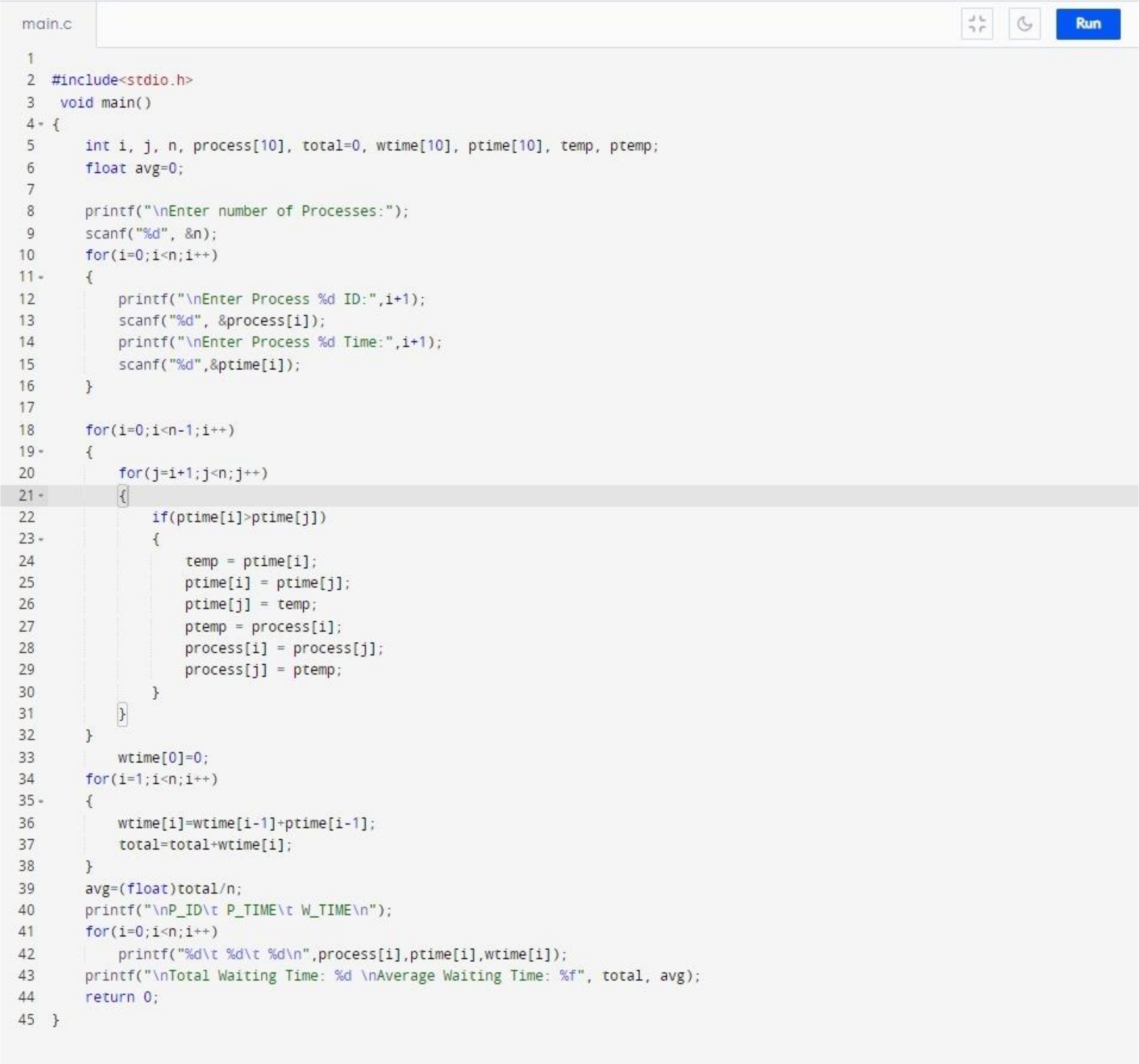
```c
        }
        else
        {
            count[i - 1]++;
        }
    }

    fclose(fptr);

    printf("\nOccurrences of all distinct words in file: \n");
    for (i=0; i<index; i++)
    {

        printf("%-15s => %d\n", words[i], count[i]);
    }

    return 0;

}
```

Output:



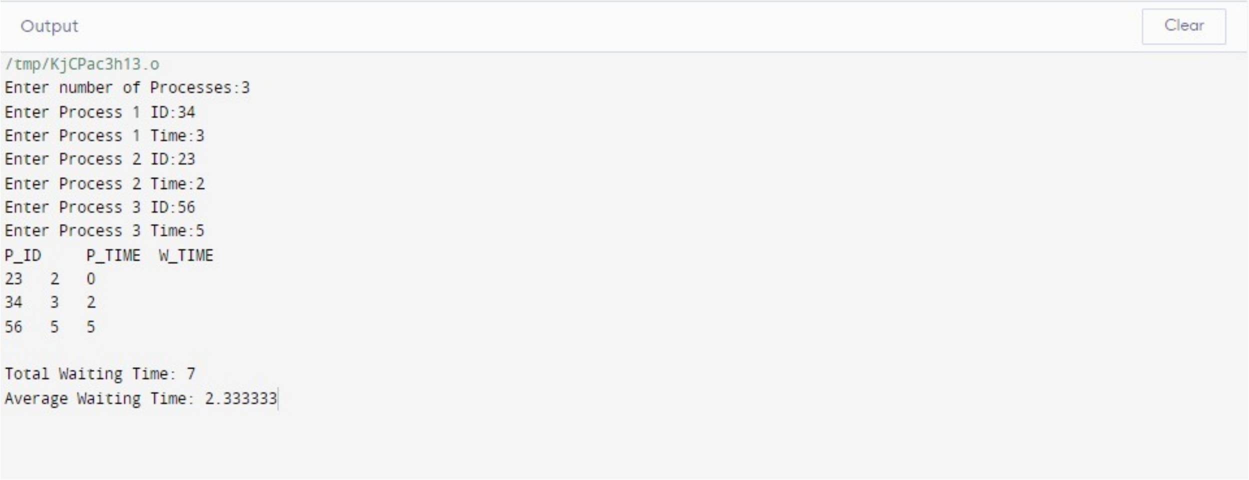
```
Occurrences of all distinct words in file:
learn           => 2
your            => 2
rules           => 2
you             => 1
better          => 1
```

Q2. Write a C Program for Shortest Job First Scheduling.



```c

#include<stdio.h>
 void main()
{
    int i, j, n, process[10], total=0, wtime[10], ptime[10], temp, ptemp;
    float avg=0;

    printf("\nEnter number of Processes:");
    scanf("%d", &n);
    for(i=0;i<n;i++)
    {
        printf("\nEnter Process %d ID:",i+1);
        scanf("%d", &process[i]);
        printf("\nEnter Process %d Time:",i+1);
        scanf("%d",&ptime[i]);
    }

    for(i=0;i<n-1;i++)
    {
        for(j=i+1;j<n;j++)
        {
            if(ptime[i]>ptime[j])
            {
                temp = ptime[i];
                ptime[i] = ptime[j];
                ptime[j] = temp;
                ptemp = process[i];
                process[i] = process[j];
                process[j] = ptemp;
            }
        }
    }
        wtime[0]=0;
    for(i=1;i<n;i++)
    {
        wtime[i]=wtime[i-1]+ptime[i-1];
        total=total+wtime[i];
    }
    avg=(float)total/n;
    printf("\nP_ID\t P_TIME\t W_TIME\n");
    for(i=0;i<n;i++)
        printf("%d\t %d\t %d\n",process[i],ptime[i],wtime[i]);
    printf("\nTotal Waiting Time: %d \nAverage Waiting Time: %f", total, avg);
    return 0;
}
```

# Output:



```
/tmp/KjCPac3h13.o
Enter number of Processes:3
Enter Process 1 ID:34
Enter Process 1 Time:3
Enter Process 2 ID:23
Enter Process 2 Time:2
Enter Process 3 ID:56
Enter Process 3 Time:5
P_ID     P_TIME  W_TIME
23   2   0
34   3   2
56   5   5

Total Waiting Time: 7
Average Waiting Time: 2.333333
```