Course Name: Operating systems

LAB: 06

Submitted By: Ebaad Khan

Roll: DT-22045

PROGRAM:

```
#include <stdio.h>


#define n 4


int compltedPhilo = 0, i;


struct fork {
   int taken;
} ForkAvil[n];


struct philosp {
   int left;
   int right;
} Philostatus[n];


void goForDinner(int philID) {
```

```
  if (Philostatus[philID].left == 10 && Philostatus[philID].right == 10) {
    // Already completed dinner
    printf("Philosopher %d already completed dinner\n", philID + 1);
  } else if (Philostatus[philID].left == 1 && Philostatus[philID].right == 1) {
    // Has both forks, completing dinner now
    printf("Philosopher %d completed his dinner\n", philID + 1);
    Philostatus[philID].left = Philostatus[philID].right = 10; // mark done

    int otherFork = philID - 1;
    if (otherFork == -1)
      otherFork = n - 1;

    ForkAvil[philID].taken = ForkAvil[otherFork].taken = 0; // release forks
    printf("Philosopher %d released fork %d and fork %d\n", philID + 1, philID + 1,
otherFork + 1);
    compltedPhilo++;
  } else if (Philostatus[philID].left == 1 && Philostatus[philID].right == 0) {
    // Has left fork, trying for right fork
    if (philID == n - 1) {
      if (ForkAvil[philID].taken == 0) {
        ForkAvil[philID].taken = 1;
        Philostatus[philID].right = 1;
        printf("Fork %d taken by philosopher %d\n", philID + 1, philID + 1);
      } else {
        printf("Philosopher %d is waiting for fork %d\n", philID + 1, philID + 1);
      }
```

```
    } else {
        int dupPhilID = philID;
        philID -= 1;
        if (philID == -1)
            philID = n - 1;

        if (ForkAvil[philID].taken == 0) {
            ForkAvil[philID].taken = 1;
            Philostatus[dupPhilID].right = 1;
            printf("Fork %d taken by Philosopher %d\n", philID + 1, dupPhilID + 1);
        } else {
            printf("Philosopher %d is waiting for Fork %d\n", dupPhilID + 1, philID + 1);
        }
    }
} else if (Philostatus[philID].left == 0) {
    // Trying to take left fork
    if (philID == n - 1) {
        if (ForkAvil[philID - 1].taken == 0) {
            ForkAvil[philID - 1].taken = 1;
            Philostatus[philID].left = 1;
            printf("Fork %d taken by philosopher %d\n", philID, philID + 1);
        } else {
            printf("Philosopher %d is waiting for fork %d\n", philID + 1, philID);
        }
    } else {
```

```
        if (ForkAvil[philID].taken == 0) {
            ForkAvil[philID].taken = 1;
            Philostatus[philID].left = 1;
            printf("Fork %d taken by Philosopher %d\n", philID + 1, philID + 1);
        } else {
            printf("Philosopher %d is waiting for Fork %d\n", philID + 1, philID + 1);
        }
    }
  }
}

int main() {
  for (i = 0; i < n; i++) {
    ForkAvil[i].taken = 0;
    Philostatus[i].left = 0;
    Philostatus[i].right = 0;
  }

  while (compltedPhilo < n) {
    for (i = 0; i < n; i++) {
      goForDinner(i);
    }
    printf("\nTill now, number of philosophers completed dinner: %d\n\n",
compltedPhilo);
  }
```

```
    return 0;
}
```

Output: