**Course Name: Operating systems**

**LAB: 06**

**Talal Khan**

**Roll: DT-22043**

PROGRAM:

```
#include <stdio.h>


#deflne 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)
```

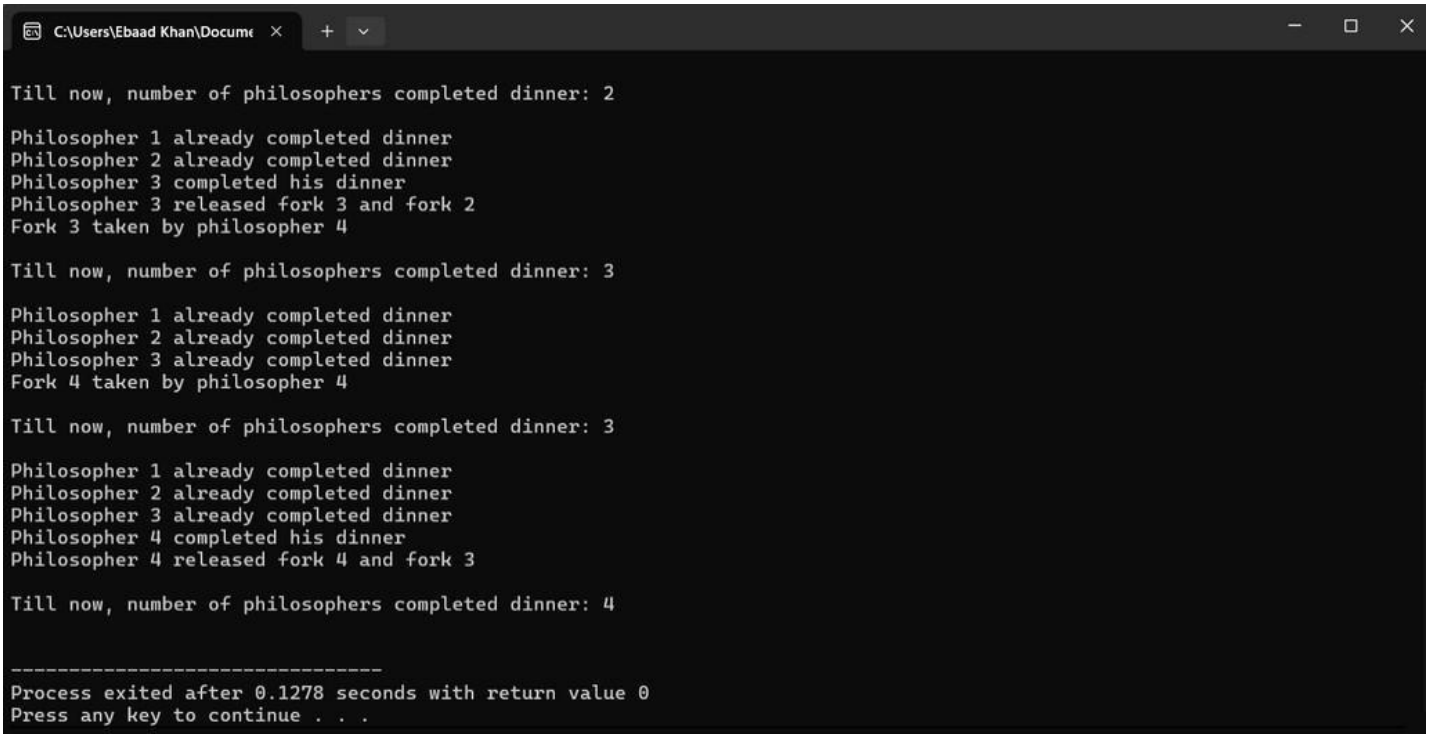
```
            philID = n - 1;

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:


C:\Users\Ebaad Khan\Docume
Till now, number of philosophers completed dinner: 2
Philosopher 1 already completed dinner
Philosopher 2 already completed dinner
Philosopher 3 completed his dinner
Philosopher 3 released fork 3 and fork 2
Fork 3 taken by philosopher 4
Till now, number of philosophers completed dinner: 3
Philosopher 1 already completed dinner
Philosopher 2 already completed dinner
Philosopher 3 already completed dinner
Fork 4 taken by philosopher 4
Till now, number of philosophers completed dinner: 3
Philosopher 1 already completed dinner
Philosopher 2 already completed dinner
Philosopher 3 already completed dinner
Philosopher 4 completed his dinner
Philosopher 4 released fork 4 and fork 3
Till now, number of philosophers completed dinner: 4
--------------------------------
Process exited after 0.1278 seconds with return value 0
Press any key to continue . . .