|  |  |  |  |
| --- | --- | --- | --- |
| SVKM's-IOT, DhuleShri Vile Parle Kelavani Mandal's  INSTITUTE OF TECHNOLOGY  DHULE (M.S.)  DEPARMENT OF COMPUTER ENGINEERING | | | |
| Subject: Competitive Programming Lab (BTCOL606) | | | Remark |
| Name : Nisha Sharad Bagul | | Roll No. : 03 |
| Class : T.Y. | Batch : T1 | Division: |
| Expt. No. :09 | Date : | | Signature |
| Title : Write a Program to implement the longest nap Problem | | |
|  | | |
|  | | |

**Code:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define START\_DAY 600 // 10:00 AM in minutes

#define END\_DAY 1080 // 6:00 PM in minutes

#define MAX 105

// Structure to store appointments

typedef struct {

int start;

int end;

} Appointment;

// Convert hh:mm to minutes

int to\_minutes(char time[]) {

int h, m;

sscanf(time, "%d:%d", &h, &m);

return h \* 60 + m;

}

// Convert minutes to hh:mm format

void to\_time\_format(int minutes, char result[]) {

int h = minutes / 60;

int m = minutes % 60;

sprintf(result, "%02d:%02d", h, m);

}

// Compare function for qsort

int compare(const void \*a, const void \*b) {

return ((Appointment\*)a)->start - ((Appointment\*)b)->start;

}

int main() {

int s, day = 1;

while (scanf("%d", &s) != EOF) {

getchar(); // consume newline

Appointment apps[MAX];

for (int i = 0; i < s; i++) {

char time1[6], time2[6], line[260];

scanf("%s %s", time1, time2);

apps[i].start = to\_minutes(time1);

apps[i].end = to\_minutes(time2);

fgets(line, 255, stdin); // read rest of line (description)

}

// Sort appointments by start time

qsort(apps, s, sizeof(Appointment), compare);

int max\_nap = 0, nap\_start = START\_DAY;

// First nap: from 10:00 to first appointment

if (apps[0].start - START\_DAY > max\_nap) {

max\_nap = apps[0].start - START\_DAY;

nap\_start = START\_DAY;

}

// Naps between appointments

for (int i = 0; i < s - 1; i++) {

int gap = apps[i + 1].start - apps[i].end;

if (gap > max\_nap) {

max\_nap = gap;

nap\_start = apps[i].end;

}

}

// Last nap: from last appointment to 18:00

if (END\_DAY - apps[s - 1].end > max\_nap) {

max\_nap = END\_DAY - apps[s - 1].end;

nap\_start = apps[s - 1].end;

}

// Print output

char start\_time[6];

to\_time\_format(nap\_start, start\_time);

printf("Day #%d: the longest nap starts at %s and will last for ", day++, start\_time);

if (max\_nap >= 60)

printf("%d hours and %d minutes.\n", max\_nap / 60, max\_nap % 60);

else

printf("%d minutes.\n", max\_nap);

}

return 0;

}

**Output:**

