```c


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

struct Employee {
    int* ratings;
    int total_score;
};

void inputEmployees(struct Employee* employees, int numEmployees, int numPeriods) {
    for (int i = 0; i < numEmployees; i++) {
        employees[i].ratings = (int*)malloc(numPeriods * sizeof(int));
        printf("Enter ratings for employee %d:\n", i + 1);
        for (int j = 0; j < numPeriods; j++) {
            printf("Period %d: ", j + 1);
            scanf("%d", &employees[i].ratings[j]);
        }
    }
}

void displayPerformance(struct Employee* employees, int numEmployees, int numPeriods) {
    for (int i = 0; i < numEmployees; i++) {
        printf("Ratings for employee %d:\n", i + 1);
        for (int j = 0; j < numPeriods; j++) {
            printf("Period %d: %d\n", j + 1, employees[i].ratings[j]);
        }
    }
}

int findEmployeeOfYear(struct Employee* employees, int numEmployees, int numPeriods) {
    int max = employees[0].ratings[0];
    int employeeIndex = 0;
    for (int i = 0; i < numEmployees; i++) {
        for (int j = 0; j < numPeriods; j++) {
            if (employees[i].ratings[j] > max) {
                max = employees[i].ratings[j];
                employeeIndex = i;
            }
        }
    }
    return employeeIndex;
}

int main() {
    int numEmployees, numPeriods;
    printf("Enter number of employees: ");
    scanf("%d", &numEmployees);
    printf("Enter number of periods: ");
    scanf("%d", &numPeriods);

    struct Employee* employees = (struct Employee*)malloc(numEmployees * sizeof(struct

    inputEmployees(employees, numEmployees, numPeriods);
    displayPerformance(employees, numEmployees, numPeriods);

    int employeeOfYear = findEmployeeOfYear(employees, numEmployees, numPeriods);
    printf("Employee of the year: %d\n", employeeOfYear + 1);

    // Free allocated memory
    for (int i = 0; i < numEmployees; i++) {
        free(employees[i].ratings)
        }
    free(employees);
    return 0;
}
```

```c

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

typedef struct {
    int* ratings;
    int totalScore;
} Employee;
void inputRatings(Employee* employee, int numPeriods) {
    employee->ratings = malloc(numPeriods * sizeof(int));
    for (int i = 0; i < numPeriods; i++) {
        printf("Enter rating for period %d: ", i + 1);
        scanf("%d", &employee->ratings[i]);
        employee->totalScore += employee->ratings[i];
    }
}
void displayPerformance(Employee* employee, int numPeriods) {
    printf("Ratings: ");
    for (int i = 0; i < numPeriods; i++) {
        printf("%d ", employee->ratings[i]);
    }
    printf("\nTotal Score: %d\n", employee->totalScore);
}
int main() {
    int numEmployees, numPeriods;
    printf("Enter number of employees: ");
    scanf("%d", &numEmployees);
    printf("Enter number of evaluation periods: ");
    scanf("%d", &numPeriods);

    Employee* employees = malloc(numEmployees * sizeof(Employee));
    for (int i = 0; i < numEmployees; i++) {
        printf("Employee %d:\n", i + 1);
        inputRatings(&employees[i], numPeriods);
        displayPerformance(&employees[i], numPeriods);

    for (int i = 0; i < numEmployees; i++) {
        free(employees[i].ratings);
    }
    free(employees);

    return 0;
}
```

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

int validateEmail(char* email) {
    int atCount = 0, dotCount = 0;
    for (int i = 0; email[i]; i++) {
        if (email[i] == '@') atCount++;
        else if (email[i] == '.' && atCount == 1) dotCount++;
    }
    return atCount == 1 && dotCount > 0;
}

int main() {
    char* email = (char*)malloc(100 * sizeof(char));
    printf("Enter an email address: ");
    fgets(email, 100, stdin);
    email[strcspn(email, "\n")] = 0;

    if (validateEmail(email)) printf("Valid Email\n");
    else printf("Invalid Email\n");

    free(email);
    return 0;
}
```

```c
#include <stdio.h>
#include <string.h>

#define BALLS 12

struct Player {
    char playerName[50];
    int ballScores[BALLS];
    int totalScore;
};

int validateScore(int score) {
    return (score >= 0 && score <= 6);
}

void playGame(struct Player *player) {
    printf("Enter scores for %s:\n", player->playerName);
    player->totalScore = 0;
    for (int i = 0; i < BALLS; i++) {
        int score;
        printf("Ball %d: ", i + 1);
        scanf("%d", &score);
        if (validateScore(score)) {
            player->ballScores[i] = score;
            player->totalScore += score;
        } else {
            printf("Invalid score! Ball is marked, but no runs added.\n");
            player->ballScores[i] = 0;
        }
    }
}
void findWinner(struct Player player1, struct Player player2) {
    printf("\nMatch Result:\n");
    if (player1.totalScore > player2.totalScore) {
        printf("%s wins with %d runs!\n", player1.playerName, player1.totalScore);
    } else if (player2.totalScore > player1.totalScore) {
        printf("%s wins with %d runs!\n", player2.playerName, player2.totalScore);
    } else {
        printf("It's a tie! Both players scored %d runs.\n", player1.totalScore);
    }
}
void displayMatchScoreboard(struct Player player1, struct Player player2) {
    printf("\n%s's Performance:\n", player1.playerName);
    for (int i = 0; i < BALLS; i++) printf("%d ", player1.ballScores[i]);
    printf("\nTotal Score: %d\n", player1.totalScore);
    printf("Average Score: %.2f\n", (double)player1.totalScore / BALLS);
    printf("\n%s's Performance:\n", player2.playerName);
    for (int i = 0; i < BALLS; i++) printf("%d ", player2.ballScores[i]);
    printf("\nTotal Score: %d\n", player2.totalScore);
    printf("Average Score: %.2f\n", (double)player2.totalScore / BALLS);
}
int main() {
    struct Player player1, player2;
    printf("Enter Player 1's name: ");
    fgets(player1.playerName, sizeof(player1.playerName), stdin);
    player1.playerName[strcspn(player1.playerName, "\n")] = 0;
    printf("Enter Player 2's name: ");
    fgets(player2.playerName, sizeof(player2.playerName), stdin);
    player2.playerName[strcspn(player2.playerName, "\n")] = 0;
    playGame(&player1);
    playGame(&player2);
    displayMatchScoreboard(player1, player2);
    findWinner(player1, player2);

    return 0;
```

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct {
    int* ratings;
    int totalScore;
} Employee;

void inputRatings(Employee* employee, int numPeriods) {
    employee->ratings = malloc(numPeriods * sizeof(int));
    for (int i = 0; i < numPeriods; i++) {
        printf("Enter rating for period %d: ", i + 1);
        scanf("%d", &employee->ratings[i]);
        employee->totalScore += employee->ratings[i];
    }
}

int main() {
    int numEmployees, numPeriods;
    printf("Enter number of employees: ");
    scanf("%d", &numEmployees);
    printf("Enter number of evaluation periods: ");
    scanf("%d", &numPeriods);

    Employee* employees = malloc(numEmployees * sizeof(Employee));

    for (int i = 0; i < numEmployees; i++) {
        inputRatings(&employees[i], numPeriods);
    }

    for (int i = 0; i < numEmployees; i++) {
        free(employees[i].ratings);
    }
    free(employees);

    return 0;
}
```

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct {
    char type[50];
} Crop;

typedef struct {
    int temperature;
} Weather;

typedef struct {
    char type[50];
} Equipment;

typedef struct {
    int soilNutrients;
} Sensor;

typedef struct {
    Crop* crops;
    Weather* weatherForecasts;
    Equipment* equipment;
    Sensor* sensorData;
} Field;

typedef struct {
    Field** fields;
} RegionalHub;

int main() {
    RegionalHub hub;
    hub.fields = malloc(2 * sizeof(Field*));

    for (int i = 0; i < 2; i++) {
        hub.fields[i] = malloc(sizeof(Field));
        hub.fields[i]->crops = malloc(sizeof(Crop));
        hub.fields[i]->weatherForecasts = malloc(sizeof(Weather));
        hub.fields[i]->equipment = malloc(sizeof(Equipment));
        hub.fields[i]->sensorData = malloc(sizeof(Sensor));
    }

    for (int i = 0; i < 2; i++) {
        free(hub.fields[i]->crops);
        free(hub.fields[i]->weatherForecasts);
        free(hub.fields[i]->equipment);
        free(hub.fields[i]->sensorData);
        free(hub.fields[i]);
    }
    free(hub.fields);

    return 0;
}
```

```c


#include <stdio.h>
#include <stdlib.h>

typedef struct {
int* scores;
} Profile;

typedef struct {
int resolution;
int playbackPosition;
} Device;

typedef struct {
char title[50];
int rating;
} Content;

int main() {
int numRows = 2;
int numCols = 3;
int** engagement = malloc(numRows * sizeof(int*));
for (int i = 0; i < numRows; i++) engagement[i] = malloc(numCols * sizeof(int));

Profile* profiles = malloc(numRows * sizeof(Profile));
for (int i = 0; i < numRows; i++) profiles[i].scores = engagement[i];

int numDevices = 2;
Device** devices = malloc(numRows * sizeof(Device*));
for (int i = 0; i < numRows; i++) devices[i] = malloc(numDevices * sizeof(Device));

int numContents = 3;
Content** contents = malloc(numCols * sizeof(Content*));
for (int i = 0; i < numCols; i++) contents[i] = malloc(numContents * sizeof(Content));

for (int i = 0; i < numRows; i++) {
    free(engagement[i]);
    free(devices[i]);
}
free(engagement);
free(devices);
for (int i = 0; i < numCols; i++) free(contents[i]);
free(contents);
free(profiles);

return 0;
```