1. #include <stdio.h>  
#define max\_users 500000  
void main() {  
    int current\_users;  
    printf("Enter the number of concurrent users: ");  
    scanf("%d", &current\_users);  
    if (current\_users > max\_users)  
    {  
        printf("Platform crashed", max\_users);  
    } else  
    {  
        printf("Platform is operating perfectly\n", current\_users);  
    }  
}

2. #include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
int main() {  
    int failedRecom = 0;  
    int totalRecom = 100;  
    float failureProb = 0.02;  
    srand(time(NULL));  
    for (int i = 0; i < totalRecom; i++) {  
        float randomValue = (float)rand() / RAND\_MAX;  
        if (randomValue < failureProb) {  
            failedRecom++;  
        }  
    }  
    printf("Number of failed recommendations: %d\n", failedRecom);  
    return 0;  
}

4. #include<stdio.h>  
#define total\_line 1000000  
#define technical\_debt 0.1  
#define reductin\_rate 0.02  
#define miteration 10  
void main()  
{  
    int total=total\_line \*technical\_debt;  
    int iteration=0;  
    printf("Total=%d",&total);  
    while(total>0&& iteration<miteration){  
        printf("Iteration %d:Remaining technical debt:%d\n",iteration,total);  
    total-=total\*reductin\_rate;  
    iteration++;  
    }  
    printf("Technical debt :%d",iteration);  
}

5. #include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
#define num\_stages 5  
  
void simulate\_order\_fulfillment(int stages[num\_stages], int num\_orders) {  
    int total\_time = 0;  
  
    for (int i = 0; i < num\_orders; i++) {  
        int order\_time = 0;  
        printf("\nOrder %d fulfillment process:\n", i + 1);  
         
        for (int j = 0; j < num\_stages; j++) {  
            printf("Stage %d: Time delay = %d seconds\n", j + 1, stages[j]);  
            order\_time += stages[j];  
        }  
         
        total\_time += order\_time;  
        printf("Total fulfillment time for Order : %d seconds\n", i + 1, order\_time);  
    }  
  
    printf("\nTotal time for fulfilling orders: %d seconds\n", num\_orders, total\_time);  
}  
  
void optimize\_order\_fulfillment(int stages[num\_stages]) {  
    int max\_time = 0;  
    int max\_stage = -1;  
  
    for (int i = 0; i < num\_stages; i++) {  
        if (stages[i] > max\_time) {  
            max\_time = stages[i];  
            max\_stage = i;  
        }  
    }  
  
    if (max\_stage != -1) {  
        printf("\nOptimizing... Reducing time delay in Stage %d by 20%%.\n", max\_stage + 1);  
        stages[max\_stage] = stages[max\_stage] - (stages[max\_stage] / 5);  
    }  
}  
  
int main() {  
    int stages[num\_stages] = {30, 40, 50, 60, 70};  
    int num\_orders;  
  
    srand(time(NULL));  
  
    printf("Enter the number of orders to simulate: ");  
    scanf("%d", &num\_orders);  
  
    printf("\nOrder Fulfillment Simulation (Before Optimization):\n");  
    simulate\_order\_fulfillment(stages, num\_orders);  
  
    optimize\_order\_fulfillment(stages);  
  
    printf("\nOrder Fulfillment Simulation (After Optimization):\n");  
    simulate\_order\_fulfillment(stages, num\_orders);  
  
    return 0;

6. #include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
#define num\_stages 5  
  
void simulate\_order\_fulfillment(int stages[num\_stages], int num\_orders) {  
    int total\_time = 0;  
  
    for (int i = 0; i < num\_orders; i++) {  
        int order\_time = 0;  
        printf("\nOrder %d fulfillment process:\n", i + 1);  
         
        for (int j = 0; j < num\_stages; j++) {  
            printf("Stage %d: Time delay = %d seconds\n", j + 1, stages[j]);  
            order\_time += stages[j];  
        }  
         
        total\_time += order\_time;  
        printf("Total fulfillment time for Order : %d seconds\n", i + 1, order\_time);  
    }  
  
    printf("\nTotal time for fulfilling orders: %d seconds\n", num\_orders, total\_time);  
}  
  
void optimize\_order\_fulfillment(int stages[num\_stages]) {  
    int max\_time = 0;  
    int max\_stage = -1;  
  
    for (int i = 0; i < num\_stages; i++) {  
        if (stages[i] > max\_time) {  
            max\_time = stages[i];  
            max\_stage = i;  
        }  
    }  
  
    if (max\_stage != -1) {  
        printf("\nOptimizing... Reducing time delay in Stage %d by 20%%.\n", max\_stage + 1);  
        stages[max\_stage] = stages[max\_stage] - (stages[max\_stage] / 5);  
    }  
}  
  
int main() {  
    int stages[num\_stages] = {30, 40, 50, 60, 70};  
    int num\_orders;  
  
    srand(time(NULL));  
  
    printf("Enter the number of orders to simulate: ");  
    scanf("%d", &num\_orders);  
  
    printf("\nOrder Fulfillment Simulation (Before Optimization):\n");  
    simulate\_order\_fulfillment(stages, num\_orders);  
  
    optimize\_order\_fulfillment(stages);  
  
    printf("\nOrder Fulfillment Simulation (After Optimization):\n");  
    simulate\_order\_fulfillment(stages, num\_orders);  
  
    return 0;  
}