**Assignment No. 07**

**Problem Statement** : Write a program to simulate memory allocation techniques: First Fit, Best Fit, Next Fit and Worst Fit.

**Code :**

#include<stdio.h>

#include<stdlib.h>

int M;

int N;

int Holes[10];

int Process[10];

void FirstFit() {

int CopyHoles[10];

int CopyProcess[10];

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

CopyHoles[i] = Holes[i];

}

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

CopyProcess[i] = Process[i];

}

int index = 0;

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

int found = 0;

for(int j = 0; j < M; j++) {

if(CopyHoles[j] >= CopyProcess[i]) {

found = 1;

index = j;

break;

}

}

if(found == 0) {

printf("Process %d cannot be allocated\n", i+1);

break;

}

else {

printf("Process %d allocated to hole %d || Process Size = %d || Hole Size = %d || Updated Hole Size = %d\n", i+1, index+1, CopyProcess[i], CopyHoles[index], CopyHoles[index] - CopyProcess[i]);

CopyHoles[index] -= CopyProcess[i];

}

}

}

void BestFit() {

int CopyHoles[10];

int CopyProcess[10];

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

CopyHoles[i] = Holes[i];

}

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

CopyProcess[i] = Process[i];

}

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

int index = -1;

int small = 999;

for(int j = 0; j < M; j++) {

if(CopyHoles[j] >= CopyProcess[i] && CopyHoles[j] < small) {

small = CopyHoles[j];

index = j;

}

}

if(index == -1) {

printf("Process %d cannot be allocated\n", i+1);

break;

}

else {

printf("Process %d allocated to hole %d || Process Size = %d || Hole Size = %d || Updated Hole Size = %d\n", i+1, index+1, CopyProcess[i], CopyHoles[index], CopyHoles[index] - CopyProcess[i]);

CopyHoles[index] -= CopyProcess[i];

}

}

}

void WorstFit() {

int CopyHoles[10];

int CopyProcess[10];

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

CopyHoles[i] = Holes[i];

}

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

CopyProcess[i] = Process[i];

}

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

int index = -1;

int large = -999;

for(int j = 0; j < M; j++) {

if(CopyHoles[j] >= CopyProcess[i] && CopyHoles[j] > large) {

large = CopyHoles[j];

index = j;

}

}

if(index == -1) {

printf("Process %d cannot be allocated\n", i+1);

break;

}

else {

printf("Process %d allocated to hole %d || Process Size = %d || Hole Size = %d || Updated Hole Size = %d\n", i+1, index+1, CopyProcess[i], CopyHoles[index], CopyHoles[index] - CopyProcess[i]);

CopyHoles[index] -= CopyProcess[i];

}

}

}

void NextFit() {

int CopyHoles[10];

int CopyProcess[10];

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

CopyHoles[i] = Holes[i];

}

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

CopyProcess[i] = Process[i];

}

int index = 0;

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

int found = 0;

int count = 0;

for(int j = index; count < M; j = (j + 1) % M) {

if(CopyHoles[j] >= CopyProcess[i]) {

index = j;

found = 1;

break;

}

count++;

}

if(found == 0) {

printf("Process %d cannot be allocated\n", i+1);

break;

}

else {

printf("Process %d allocated to hole %d || Process Size = %d || Hole Size = %d || Updated Hole Size = %d\n", i+1, index+1, CopyProcess[i], CopyHoles[index], CopyHoles[index] - CopyProcess[i]);

CopyHoles[index] -= CopyProcess[i];

}

}

}

int main() {

int choice;

printf("Enter the No. of Holes(Max = 10): ");

scanf("%d", &M);

printf("Enter the No. of Processes(Max = 10): ");

scanf("%d", &N);

printf("Enter the Hole Size one by one:\n");

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

scanf("%d", &Holes[i]);

}

printf("Enter the Process Size one by one:\n");

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

scanf("%d", &Process[i]);

}

do {

printf("\n\*\*\*\*\*\*\*\*\*\*Menu\*\*\*\*\*\*\*\*\*\*\n");

printf("1. First Fit\n2. Best Fit\n3. Worst Fit\n4. Next Fit\n5. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch(choice) {

case 1: {

printf("\nFirst Fit Allocation\n");

FirstFit();

printf("-----------------------------------------------------------------\n");

break; }

case 2: {

printf("\nBest Fit Allocation\n");

BestFit();

printf("-----------------------------------------------------------------\n");

break; }

case 3: {

printf("\nWorst Fit Allocation\n");

WorstFit();

printf("-----------------------------------------------------------------\n");

break; }

case 4: {

printf("\nNext Fit Allocation\n");

NextFit();

printf("-----------------------------------------------------------------\n");

break; }

case 5: {

printf("Exiting...\n");

break; }

default: {

printf("Invalid choice. Please try again.\n");

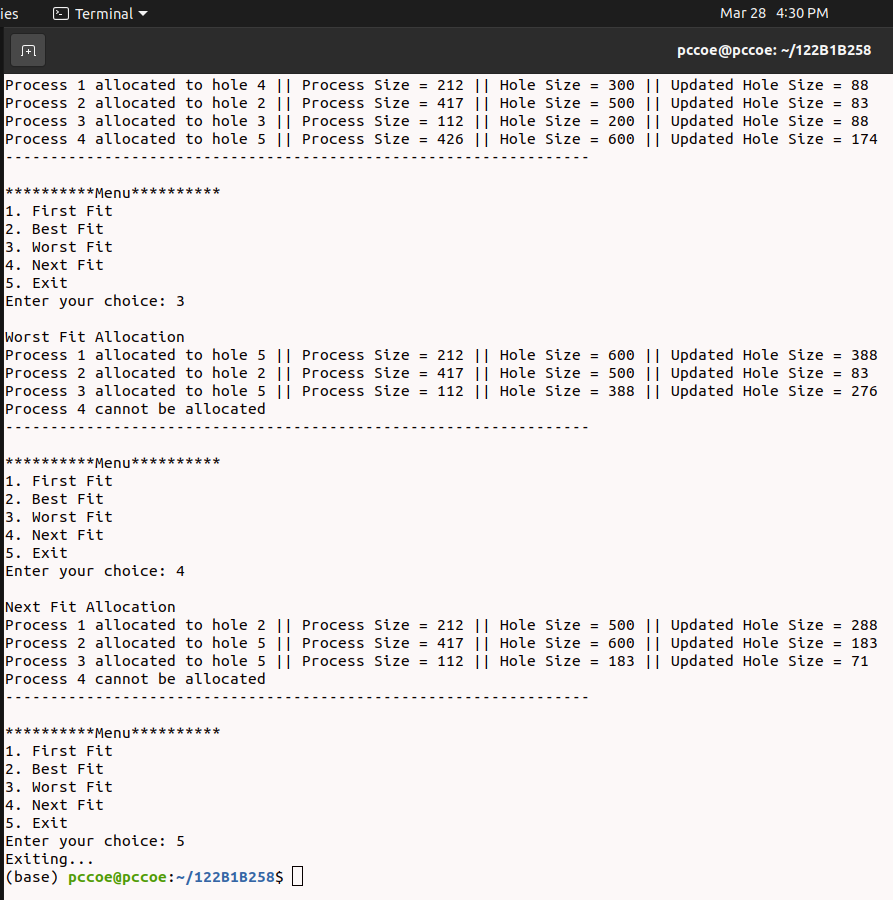
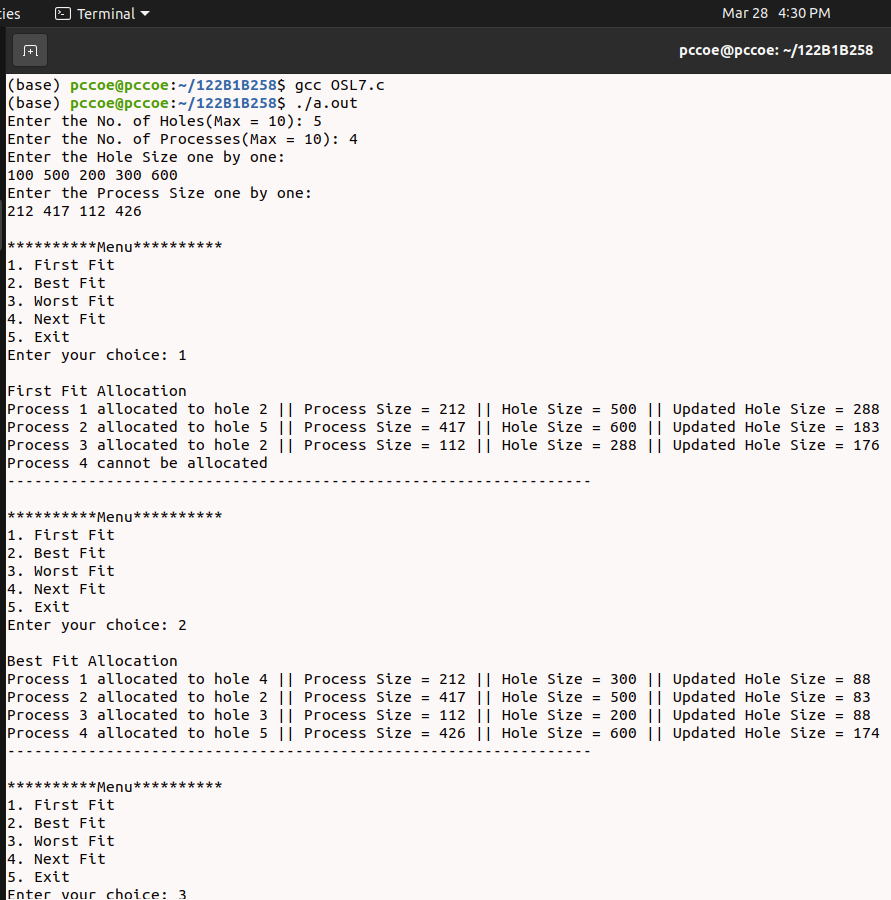
break; }

}

} while(choice != 5);

return 0;

}

**Output :**