**Week - 7**

**(03 August 2023)**

**Experiment - 7**

**Question:**

Write a C program to simulate the following contiguous memory allocation techniques:

1. Worst-fit
2. Best-fit

**(c)** First-fit

**Program:**

#include <stdio.h>

#define max 25

void firstFit(int b[], int nb, int f[], int nf);

void worstFit(int b[], int nb, int f[], int nf);

void bestFit(int b[], int nb, int f[], int nf);

int main()

{

int b[max], f[max], nb, nf;

printf("Memory Management Schemes\n");

printf("\nEnter the number of blocks:");

scanf("%d", &nb);

printf("Enter the number of files:");

scanf("%d", &nf);

printf("\nEnter the size of the blocks:\n");

for (int i = 1; i <= nb; i++)

{

printf("Block %d:", i);

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

}

printf("\nEnter the size of the files:\n");

for (int i = 1; i <= nf; i++)

{

printf("File %d:", i);

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

}

printf("\nMemory Management Scheme - First Fit");

firstFit(b, nb, f, nf);

printf("\n\nMemory Management Scheme - Worst Fit");

worstFit(b, nb, f, nf);

printf("\n\nMemory Management Scheme - Best Fit");

bestFit(b, nb, f, nf);

return 0;

}

void firstFit(int b[], int nb, int f[], int nf)

{

int bf[max] = {0};

int ff[max] = {0};

int frag[max], i, j;

for (i = 1; i <= nf; i++)

{

for (j = 1; j <= nb; j++)

{

if (bf[j] != 1 && b[j] >= f[i])

{

ff[i] = j;

bf[j] = 1;

frag[i] = b[j] - f[i];

break;

}

}

}

printf("\nFile\_no:\tFile\_size:\tBlock\_no:\tBlock\_size:\tFragment");

for (i = 1; i <= nf; i++)

{

printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i, f[i], ff[i], b[ff[i]], frag[i]);

}

}

void worstFit(int b[], int nb, int f[], int nf)

{

int bf[max] = {0};

int ff[max] = {0};

int frag[max], i, j, temp, highest = 0;

for (i = 1; i <= nf; i++)

{

for (j = 1; j <= nb; j++)

{

if (bf[j] != 1)

{

temp = b[j] - f[i];

if (temp >= 0 && highest < temp)

{

ff[i] = j;

highest = temp;

}

}

}

frag[i] = highest;

bf[ff[i]] = 1;

highest = 0;

}

printf("\nFile\_no:\tFile\_size:\tBlock\_no:\tBlock\_size:\tFragment");

for (i = 1; i <= nf; i++)

{

printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i, f[i], ff[i], b[ff[i]], frag[i]);

}

}

void bestFit(int b[], int nb, int f[], int nf)

{

int bf[max] = {0};

int ff[max] = {0};

int frag[max], i, j, temp, lowest = 10000;

for (i = 1; i <= nf; i++)

{

for (j = 1; j <= nb; j++)

{

if (bf[j] != 1)

{

temp = b[j] - f[i];

if (temp >= 0 && lowest > temp)

{

ff[i] = j;

lowest = temp;

}

}

}

frag[i] = lowest;

bf[ff[i]] = 1;

lowest = 10000;

}

printf("\nFile\_no:\tFile\_size:\tBlock\_no:\tBlock\_size:\tFragment");

for (i = 1; i <= nf && ff[i] != 0; i++)

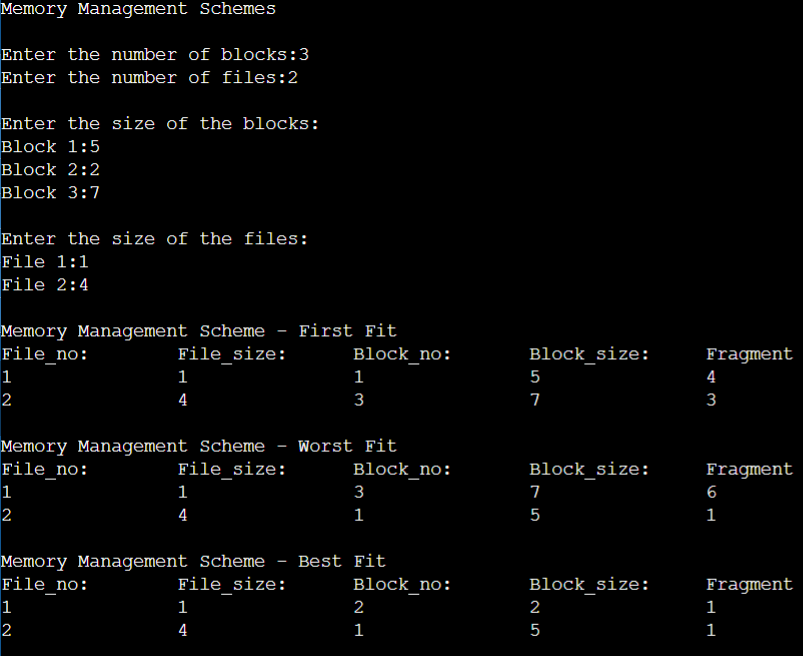
{

printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i, f[i], ff[i], b[ff[i]], frag[i]);

}

}

**Output:**



**Observation Book Pictures:**

