Lab-10

|  |
| --- |
| #include <stdio.h>  int main() {  int p[10], np, b[10], nb, ch, c[10], d[10], alloc[10], flag[10], i, j;  printf("Enter the number of processes: ");  scanf("%d", &np);  printf("Enter the number of blocks: ");  scanf("%d", &nb);  printf("Enter the size of each process:\n");  for (i = 0; i < np; i++) {  printf("Process %d: ", i);  scanf("%d", &p[i]);  }  printf("Enter the size of each block:\n");  for (j = 0; j < nb; j++) {  printf("Block %d: ", j);  scanf("%d", &b[j]);  c[j] = b[j]; // copy for best fit  d[j] = b[j]; // copy for worst fit  }  if (np <= nb) {  do {  printf("\n1. First Fit\n2. Best Fit\n3. Worst Fit\nEnter your choice: ");  scanf("%d", &ch);  for (i = 0; i < np; i++) flag[i] = 1; // reset flags  switch (ch) {  case 1: // First Fit  printf("\n---- First Fit ----\n");  for (i = 0; i < np; i++) {  for (j = 0; j < nb; j++) {  if (p[i] <= b[j]) {  alloc[i] = j;  printf("Process %d of size %d allocated to Block %d of size %d\n", i, p[i], j, b[j]);  b[j] = 0;  flag[i] = 0;  break;  }  }  if (flag[i])  printf("Process %d of size %d not allocated.\n", i, p[i]);  }  break;  case 2: // Best Fit  printf("\n---- Best Fit ----\n");  // Sort blocks in ascending  for (i = 0; i < nb - 1; i++)  for (j = i + 1; j < nb; j++)  if (c[i] > c[j]) {  int temp = c[i];  c[i] = c[j];  c[j] = temp;  }  for (i = 0; i < np; i++) {  for (j = 0; j < nb; j++) {  if (p[i] <= c[j]) {  printf("Process %d of size %d allocated to Block of size %d\n", i, p[i], c[j]);  c[j] = 0;  flag[i] = 0;  break;  }  }  if (flag[i])  printf("Process %d of size %d not allocated.\n", i, p[i]);  }  break;  case 3: // Worst Fit  printf("\n---- Worst Fit ----\n");  // Sort blocks in descending  for (i = 0; i < nb - 1; i++)  for (j = i + 1; j < nb; j++)  if (d[i] < d[j]) {  int temp = d[i];  d[i] = d[j];  d[j] = temp;  }  for (i = 0; i < np; i++) {  for (j = 0; j < nb; j++) {  if (p[i] <= d[j]) {  printf("Process %d of size %d allocated to Block of size %d\n", i, p[i], d[j]);  d[j] = 0;  flag[i] = 0;  break;  }  }  if (flag[i])  printf("Process %d of size %d not allocated.\n", i, p[i]);  }  break;  default:  printf("Invalid choice!\n");  }  } while (ch <= 3);  } else {  printf("Cannot allocate: More processes than blocks.\n");  }  return 0;  } |





