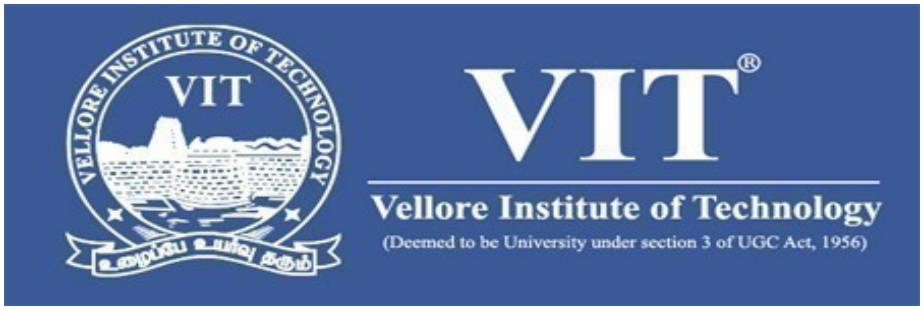

19BIT0292

Bhaumik Tandan

ASSESSMENT-5

OPERATING SYSTEM Laboratory

ITE2002

Q1. Write a program to implement the first fit, best fit, and worst fit algorithm for memory allocation.

CODE

```
#include <iostream>
#include <iomanip>
using namespace std;
typedef struct
{
    int n;
    int v;
} block;
main()
{
    cout << "1)First Fit\n2)Best Fit\n3)Worst Fit\nEnter your choice: ";
    int o;
    cin >> o;
    cout << "Enter the number of processes: ";
    int n;
    cin >> n;
    int *p = (int *)malloc(sizeof(int) * n);
    for (int i = 0; i < n; i++)
    {
        cout << "Enter the memory size for process " << i + 1 << ": ";
        cin >> p[i];
    }
    cout << "Enter the number of memory blocks: ";
    int bn;
    cin >> bn;
```

```
    block *b = (block *)malloc(sizeof(block) * bn);
    for (int i = 0; i < bn; i++)
    {
        cout << "Enter the memory size for block " << i +
1 << ": ";
        cin >> b[i].v;
        b[i].n = i + 1;
    }
    for (int i = 0; i < bn - 1; i++)
        for (int j = 0; j < bn - i - 1; j++)
            if (o == 2 && b[j].v > b[j + 1].v)
            {

                block t = b[j];
                b[j] = b[j + 1];
                b[j + 1] = t;
            }
            else if (o == 3 && b[j].v < b[j + 1].v)
            {
                block t = b[j];
                b[j] = b[j + 1];
                b[j + 1] = t;
            }
    cout << left << setw(25) << setfill(' ') << "Process
Number";
    cout << left << setw(25) << setfill(' ') << "Process
Size";
    cout << left << setw(25) << setfill(' ') << "Block
Number";
    for (int i = 0; i < n; i++)
    {
```

```
        cout << endl
             << left << setw(25) << setfill(' ') << i + 1;
        cout << left << setw(25) << setfill(' ') << p[i];
        int j;
        for (j = 0; j < bn; j++)
            if (p[i] <= b[j].v)
            {
                cout << left << setw(25) << setfill(' ')
<< b[j].n;
                b[j].v -= p[i];
                break;
            }
        if (j == bn)
            cout << left << setw(25) << setfill(' ') <<
"Not Allocated";
    }
}
```

memory_management.cpp

memory_management.cpp > main()

```
#include <iostream>
#include <iomanip>
using namespace std;
typedef struct
{
    int n;
    int v;
} block;
main()
{
    cout << "1)First Fit\n2)Best Fit\n3)Worst Fit\nEnter your choice: ";
    int o;
    cin >> o;
    cout << "Enter the number of processes: ";
    int n;
    cin >> n;
    int *p = (int *)malloc(sizeof(int) * n);
    for (int i = 0; i < n; i++)
    {
        cout << "Enter the memory size for process " << i + 1 << ": ";
        cin >> p[i];
    }
```

```
    cout << "Enter the number of memory blocks: ";
    int bn;
    cin >> bn;
    block *b = (block *)malloc(sizeof(block) * bn);
    for (int i = 0; i < bn; i++)
    {
        cout << "Enter the memory size for block " << i + 1 << ": ";
        cin >> b[i].v;
        b[i].n = i + 1;
    }
    for (int i = 0; i < bn - 1; i++)
        for (int j = 0; j < bn - i - 1; j++)
            if (o == 2 && b[j].v > b[j + 1].v)
            {

                block t = b[j];
                b[j] = b[j + 1];
                b[j + 1] = t;
            }
            else if (o == 3 && b[j].v < b[j + 1].v)
            {
                block t = b[j];
              b[j] = b[j + 1];
              b[j + 1] = t;
            }
    cout << left << setw(25) << setfill(' ') << "Process Number";
    cout << left << setw(25) << setfill(' ') << "Process Size";
    cout << left << setw(25) << setfill(' ') << "Block Number";
    for (int i = 0; i < n; i++)
    {
        cout << endl
             << left << setw(25) << setfill(' ') << i + 1;
        cout << left << setw(25) << setfill(' ') << p[i];
        int j;
        for (j = 0; j < bn; j++)
            if (p[i] <= b[j].v)
            {
                cout << left << setw(25) << setfill(' ') << b[j].n;
                b[j].v -= p[i];
                break;
            }
        if (j == bn)
            cout << left << setw(25) << setfill(' ') << "Not Allocated";
    }
}
```

OUTPUT

FIRST FIT

```
c:\Users\bhaum\Downloads>cd "c:\Users\bhaum\OneDrive\Desktop\os_
memory_management
1)First Fit
2)Best Fit
3)Worst Fit
Enter your choice: 1
Enter the number of processes: 4
Enter the memory size for process 1: 212
Enter the memory size for process 2: 417
Enter the memory size for process 3: 112
Enter the memory size for process 4: 426
Enter the number of memory blocks: 5
Enter the memory size for block 1: 100
Enter the memory size for block 2: 500
Enter the memory size for block 3: 200
Enter the memory size for block 4: 300
Enter the memory size for block 5: 600
Process Number          Process Size            Block Number
1                       212                     2
2                       417                     5
3                       112                     2
4                       426                     Not Allocated
```

BEST FIT

```
c:\Users\bhaum\OneDrive\Desktop\os_Da>cd "c:\Users\bhaum\OneDr
sktop\os_Da\"memory_management
1)First Fit
2)Best Fit
3)Worst Fit
Enter your choice: 2
Enter the number of processes: 3
Enter the memory size for process 1: 1
Enter the memory size for process 2: 4
Enter the memory size for process 3: 3
Enter the number of memory blocks: 4
Enter the memory size for block 1: 5
Enter the memory size for block 2: 8
Enter the memory size for block 3: 4
Enter the memory size for block 4: 10
Process Number          Process Size            Block Number
1                       1                       3
2                       4                       1
3                       3                       3
```

WORST FIT

```
c:\Users\bhaum\OneDrive\Desktop\os_Da>cd "c:\Users\bhaum\OneDrive
sktop\os_Da\"memory_management
1)First Fit
2)Best Fit
3)Worst Fit
Enter your choice: 3
Enter the number of processes: 4
Enter the memory size for process 1: 212
Enter the memory size for process 2: 417
Enter the memory size for process 3: 112
Enter the memory size for process 4: 426
Enter the number of memory blocks: 5
Enter the memory size for block 1: 100
Enter the memory size for block 2: 500
Enter the memory size for block 3: 200
Enter the memory size for block 4: 300
Enter the memory size for block 5: 600
Process Number          Process Size            Block Number
1                       212                     5
2                       417                     2
3                       112                     5
4                       426                     Not Allocated
```

Q2. Write a program to implement the page replacement algorithms.

a. FIFO

b. LRU

c. OPTIMAL

CODE

```
#include <iostream>
#include <iomanip>
using namespace std;
main()
{
    cout << "1)FIFO\n2)LRU\n3)OPTIMAL\nEnter your choice : ";
    int o;
    cin >> o;
    cout << "Enter the number of elements in page reference string: ";
    int n;
    cin >> n;
    int *p = (int *)malloc(sizeof(int) * n);
    cout << "Enter the page reference string: ";
    for (int i = 0; i < n; i++)
        cin >> p[i];
    cout << "Enter the number of page frames: ";
    int fn;
    cin >> fn;
    int *f = (int *)malloc(sizeof(int) * fn), *q;
    for (int i = 0; i < fn; i++)
        f[i] = -1;
    cout << "\n\n"
         << left << setw(25) << setfill(' ') << "Action";
    cout << left << setw(20) << setfill(' ') << "Before";
    cout << left << setw(20) << setfill(' ') << "After";
```

```
    int ph = 0, pf = 0, re = 0, e = 0;
    for (int i = 0; i < n; i++)
    {
        int fl = 0;
        for (int j = 0; j < fn; j++)
            if (f[j] == p[i])
            {
                fl = 1;
                cout << endl
                     << left << setw(25) << setfill(' ')
<< "Page Hit";
                ph++;
                break;
            }
        if (fl == 0)
        {
            cout << endl
                 << left << setw(25) << setfill(' ') <<
"Page Fault";
            for (int j = fn - 1; j >= 0; j--)
                cout << f[j] << " ";
            if (o == 1)
            {
                f[re] = p[i];
                re = (re + 1) % fn;
            }
            else if (o == 2)
            {
                int t = 0, b;
                for (int j = 0; j < fn; j++)
```

```
        {
            int k;
            for (k = i - 1; k >= 0; k--)
                if (f[j] == p[k])
                    break;
            if (i - k > t)
            {
                t = i - k;
                b = j;
            }
        }
        f[b] = p[i];
    }
    else
    {
        if (e < fn)
            f[e++] = p[i];
        else
        {
            int t = 0, b;
            for (int j = 0; j < fn; j++)
            {
                int k;
                for (k = i + 1; k < n; k++)
                    if (f[j] == p[k])
                        break;
                if (k - i > t)
                {
                    t = k - i;
```

```
                    b = j;
                }
            }
            f[b] = p[i];
        }
    }

    cout << left << setw(15) << setfill(' ') << "
";
    for (int j = fn - 1; j >= 0; j--)
        cout << f[j] << " ";
    pf++;
  }
 }
 cout << "\n\nTotal Page Faults: " << pf << "\nTotal
Page Hits: " << ph;
}
```

page_replacement.cpp M

page_replacement.cpp > main()

```cpp
#include <iostream>
#include <iomanip>
using namespace std;
main()
{
    cout << "1)FIFO\n2)LRU\n3)OPTIMAL\nEnter your choice : ";
    int o;
    cin >> o;
    cout << "Enter the number of elements in page reference string: ";
    int n;
    cin >> n;
    int *p = (int *)malloc(sizeof(int) * n);
    cout << "Enter the page reference string: ";
    for (int i = 0; i < n; i++)
        cin >> p[i];
    cout << "Enter the number of page frames: ";
    int fn;
    cin >> fn;
    int *f = (int *)malloc(sizeof(int) * fn), *q;
    for (int i = 0; i < fn; i++)
        f[i] = -1;
    cout << "\n\n"
        << left << setw(25) << setfill(' ') << "Action";
```

```
    cout << left << setw(20) << setfill(' ') << "Before";
    cout << left << setw(20) << setfill(' ') << "After";
    int ph = 0, pf = 0, re = 0, e = 0;
    for (int i = 0; i < n; i++)
    {
        int fl = 0;
        for (int j = 0; j < fn; j++)
            if (f[j] == p[i])
            {
                fl = 1;
                cout << endl
                     << left << setw(25) << setfill(' ') << "Page Hit";
                ph++;
                break;
            }
        if (fl == 0)
        {
            cout << endl
                 << left << setw(25) << setfill(' ') << "Page Fault";
            for (int j = fn - 1; j >= 0; j--)
                cout << f[j] << " ";
            if (o == 1)
            {
                f[re] = p[i];
                re = (re + 1) % fn;
            }
            else if (o == 2)
            {
                int t = 0, b;
                for (int j = 0; j < fn; j++)
                {
                    int k;
                    for (k = i - 1; k >= 0; k--)
                        if (f[j] == p[k])
                            break;
                    if (i - k > t)
                    {
                        t = i - k;
                        b = j;
                    }
                }
                f[b] = p[i];
            }
            else
            {
        {
            if (e < fn)
                f[e++] = p[i];
            else
            {
                int t = 0, b;
                for (int j = 0; j < fn; j++)
                {
                    int k;
                    for (k = i + 1; k < n; k++)
                        if (f[j] == p[k])
```

```
79                                  break;
80                              if (k - i > t)
81                              {
82                                  t = k - i;
83                                  b = j;
84                              }
85                          }
86                          f[b] = p[i];
87                      }
88                  }
89
90                  cout << left << setw(15) << setfill(' ') << " ";
91                  for (int j = fn - 1; j >= 0; j--)
92                      cout << f[j] << " ";
93                  pf++;
94              }
95          }
96          cout << "\n\nTotal Page Faults: " << pf << "\nTotal Page Hits: " << ph;
97      }
```

OUTPUT

FIFO

```
c:\Users\bhaum\OneDrive\Desktop\os_Da>cd "c:\Users\bhaum\OneDri
top\os_Da\"page_replacement
1)FIFO
2)LRU
3)OPTIMAL
Enter your choice : 1
Enter the number of elements in page reference string: 15
Enter the page reference string: 1 2 1 3 5 5 1 6 7 1 7 4 2 2 3
Enter the number of page frames: 3


Action                  Before              After
Page Fault              -1 -1 -1               -1 -1 1
Page Fault              -1 -1 1                -1 2 1
Page Hit
Page Fault              -1 2 1                  3 2 1
Page Fault              3 2 1                  3 2 5
Page Hit
Page Fault              3 2 5                  3 1 5
Page Fault              3 1 5                  6 1 5
Page Fault              6 1 5                  6 1 7
Page Hit
Page Hit
Page Fault              6 1 7                  6 4 7
Page Fault              6 4 7                  2 4 7
Page Hit
Page Fault              2 4 7                  2 4 3

Total Page Faults: 10
Total Page Hits: 5
```

LRU

```
c:\Users\bhaum\OneDrive\Desktop\os_Da>cd "c:\Users\bhaum\OneDriv
top\os_Da\"page_replacement
1)FIFO
2)LRU
3)OPTIMAL
Enter your choice : 2
Enter the number of elements in page reference string: 15
Enter the page reference string: 1 2 1 3 5 5 1 6 7 1 7 4 2 2 3
Enter the number of page frames: 4


Action                  Before              After
Page Fault              -1 -1 -1 -1                -1 -1 -1 1
Page Fault              -1 -1 -1 1                -1 -1 2 1
Page Hit
Page Fault              -1 -1 2 1                -1 3 2 1
Page Fault              -1 3 2 1                5 3 2 1
Page Hit
Page Hit
Page Fault              5 3 2 1                 5 3 6 1
Page Fault              5 3 6 1                 5 7 6 1
Page Hit
Page Hit
Page Fault              5 7 6 1                 4 7 6 1
Page Fault              4 7 6 1                 4 7 2 1
Page Hit
Page Fault              4 7 2 1                 4 7 2 3

Total Page Faults: 9
Total Page Hits: 6
```

OPTIMAL

```
c:\Users\bhaum\OneDrive\Desktop\os_Da>cd "c:\Users\bhaum\OneDrive\Desktop\os_Da\" && g++ page
top\os_Da\"page_replacement
1)FIFO
2)LRU
3)OPTIMAL
Enter your choice : 3
Enter the number of elements in page reference string: 15
Enter the page reference string: 1 2 1 3 5 5 1 6 7 1 7 4 2 2 3
Enter the number of page frames: 3


Action                  Before              After
Page Fault              -1 -1 -1                -1 -1 1
Page Fault              -1 -1 1                -1 2 1
Page Hit
Page Fault              -1 2 1                 3 2 1
Page Fault              3 2 1                 5 2 1
Page Hit
Page Hit
Page Fault              5 2 1                 6 2 1
Page Fault              6 2 1                 7 2 1
Page Hit
Page Hit
Page Fault              7 2 1                 7 2 4
Page Hit
Page Hit
Page Fault              7 2 4                 7 2 3

Total Page Faults: 8
Total Page Hits: 7
```

Q3. Implement the following algorithms to perform file allocation.

a. Sequential

b. Linked

c. Indexed

CODE

```
#include <iostream>
using namespace std;
main()
{
    int *h = (int *)calloc(sizeof(int), 500);
    cout << "1)Sequential\n2)Linked\n3)Indexed\nEnter your
choice :";
    int o;
    cin >> o;
    cout << "Enter the number of files: ";
    int n;
    cin >> n;
    int **ind;
    if (o == 3)
        ind = (int **)malloc(sizeof(int *) * 500);
    int *fi = (int *)malloc(sizeof(int) * n);
```

```
    for (int i = 0; i < n; i++)
    {
        cout << "\nEnter the size of file " << i + 1 << ":
";
        int s;
        cin >> s;
        if (o == 1)
            while (true)
            {
                int j, a = rand() % (501 - s), f = 1;
                for (j = a; j < a + s; j++)
                    if (h[j] != 0)
                    {
                        f = 0;
                        break;
                    }
                if (f == 0)
                    continue;
                fi[i] = a;
                for (j = a; j < a + s; j++)
                    h[j] = i + 1;
                break;
            }
        else if (o == 2)
        {
            int *p = fi + i;
            for (int j = 0; j <= s; j++)
            {
                int b;
```

```
            while (true)
            {
                b = rand() % 500;
                if (h[b] == 0)
                    break;
            }
            *p = b;
            p = h + b;
        }
        *p = -1;
    }
    else
    {
        int in;
        while (true)
        {
            in = rand() % 500;
            if (h[in] == 0)
                break;
        }
        h[in] = s;
        fi[i] = in;
        ind[in] = (int *)malloc(sizeof(int) * s);
        for (int j = 0; j < s; j++)
        {
            while (true)
            {
                in = rand() % 500;
                if (h[in] == 0)
```

```
                    break;
                }
                h[in] = i + 1;
                ind[fi[i]][j] = in;
            }
        }
    }
    cout << "\nAssigned blocks are:-\n\n";
    for (int i = 0; i < n; i++)
    {
        int j = fi[i];
        if (o == 1)
            while (h[fi[i]] == h[j++])
                cout << j + 1 << "-->" << i + 1 << endl;
        else if (o == 2)
            while (h[j] != -1)
            {
                cout << j + 1 << "-->" << i + 1 << endl;
                j = h[j];
            }
        else
            for (int k = 0; k < h[j]; k++)
                cout << j + 1 << ":" << ind[j][k] << "-->"
<< i + 1 << endl;
        cout << endl;
    }
}
```

SCREENSHOT

file_allocation.cpp

file_allocation.cpp > main()

```cpp
#include <iostream>
using namespace std;
main()
{
    int *h = (int *)calloc(sizeof(int), 500);
    cout << "1)Sequential\n2)Linked\n3)Indexed\nEnter your choice :";
    int o;
    cin >> o;
    cout << "Enter the number of files: ";
    int n;
    cin >> n;
    int **ind;
    if (o == 3)
        ind = (int **)malloc(sizeof(int *) * 500);
    int *fi = (int *)malloc(sizeof(int) * n);
    for (int i = 0; i < n; i++)
    {
        cout << "\nEnter the size of file " << i + 1 << ": ";
        int s;
        cin >> s;
        if (o == 1)
            while (true)
            {
                int j, a = rand() % (501 - s), f = 1;
                for (j = a; j < a + s; j++)
                    if (h[j] != 0)
                    {
                        f = 0;
                        break;
                    }
                if (f == 0)
                    continue;
                fi[i] = a;
                for (j = a; j < a + s; j++)
                    h[j] = i + 1;
                break;
            }
```

```
38          else if (o == 2)
39          {
40              int *p = fi + i;
41              for (int j = 0; j <= s; j++)
42              {
43                  int b;
44                  while (true)
45                  {
46                      b = rand() % 500;
47                      if (h[b] == 0)
48                          break;
49                  }
50                  *p = b;
51                  p = h + b;
52              }
53              *p = -1;
54          }
55           else
56           {
57               int in;
58               while (true)
59               {
60                   in = rand() % 500;
61                   if (h[in] == 0)
62                       break;
63               }
64               h[in] = s;
65               fi[i] = in;
66               ind[in] = (int *)malloc(sizeof(int) * s);
67               for (int j = 0; j < s; j++)
68               {
69                   while (true)
70                   {
71                       in = rand() % 500;
72                       if (h[in] == 0)
73                           break;
74                   }
75                   h[in] = i + 1;
76                   ind[fi[i]][j] = in;
77               }
78           }
79      }
80      cout << "\nAssigned blocks are:-\n\n";
```

```
81      for (int i = 0; i < n; i++)
82      {
83          int j = fi[i];
84          if (o == 1)
85              while (h[fi[i]] == h[j++])
86                  cout << j + 1 << "-->" << i + 1 << endl;
87          else if (o == 2)
88              while (h[j] != -1)
89              {
90                  cout << j + 1 << "-->" << i + 1 << endl;
91                  j = h[j];
92              }
93          else
94              for (int k = 0; k < h[j]; k++)
95                  cout << j + 1 << ":" << ind[j][k] << "-->" << i + 1 << endl;
96          cout << endl;
97      }
98  }
```

OUTPUT

SEQUENTIAL

```
p\os_Da\"file_allocation
1)Sequential
2)Linked
3)Indexed
Enter your choice :1
Enter the number of files: 2

Enter the size of file 1: 5

Enter the size of file 2: 2

Assigned blocks are:-

159-->1
160-->1
161-->1
162-->1
163-->1

350-->2
351-->2
```

LINKED

```
c:\Users\bhaum\OneDrive\Deskt
p\os_Da\"file_allocation
1)Sequential
2)Linked
3)Indexed
Enter your choice :2
Enter the number of files: 3

Enter the size of file 1: 3

Enter the size of file 2: 2

Enter the size of file 3: 2

Assigned blocks are:-

434-->1
244-->1
263-->1

201-->2
9-->2

257-->3
```

INDEXED

```
c:\Users\bhaum\OneDrive\Desktop\os_Da>cd "c:\Users\bhaum\Or
p\os_Da\"file_allocation
1)Sequential
2)Linked
3)Indexed
Enter your choice :3
Enter the number of files: 2

Enter the size of file 1: 2

Enter the size of file 2: 2

Assigned blocks are:-

434:243-->1
434:262-->1

30:200-->2
30:8-->2
```

CLICK HERE FOR GITHHUB LINK