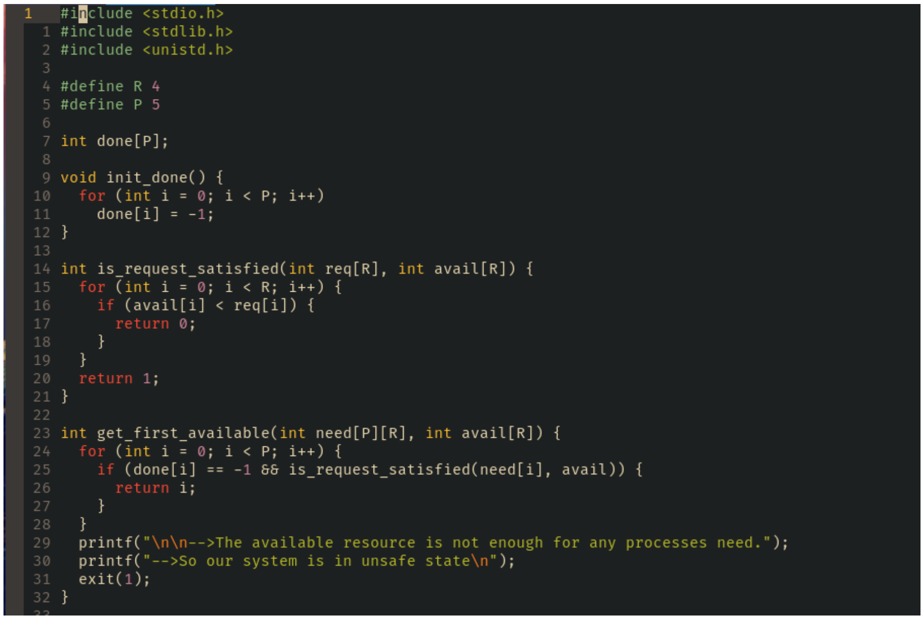
106119029 , OS Lab 9

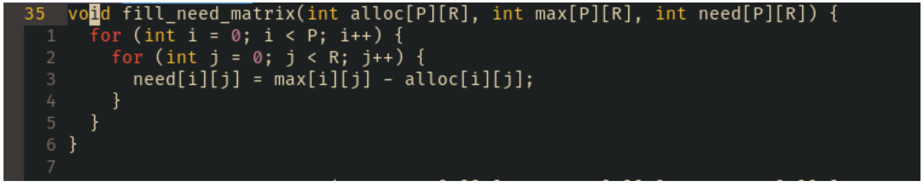
Dipesh Kafle

Banker's Algorithm

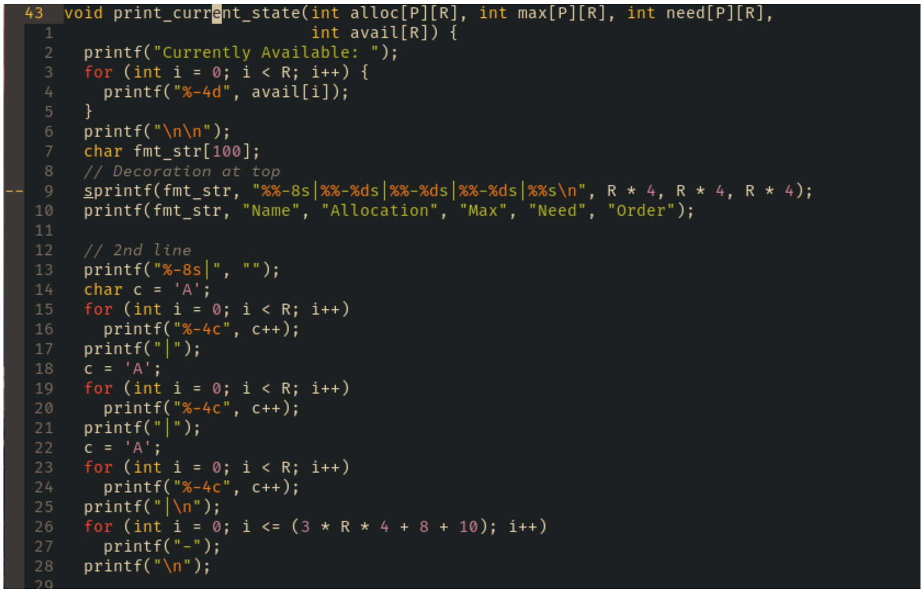
Code



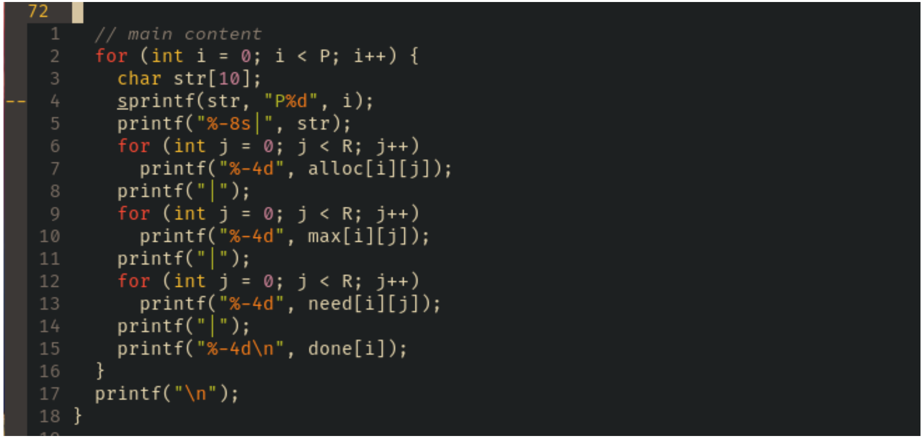
```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

#define R 4
#define P 5

int done[P];

void init_done() {
  for (int i = 0; i < P; i++)
    done[i] = -1;
}

int is_request_satisfied(int req[R], int avail[R]) {
  for (int i = 0; i < R; i++) {
    if (avail[i] < req[i]) {
      return 0;
    }
  }
  return 1;
}

int get_first_available(int need[P][R], int avail[R]) {
  for (int i = 0; i < P; i++) {
    if (done[i] == -1 && is_request_satisfied(need[i], avail)) {
      return i;
    }
  }
  printf("\n\n-->The available resource is not enough for any processes need.");
  printf("-->So our system is in unsafe state\n");
  exit(1);
}
```



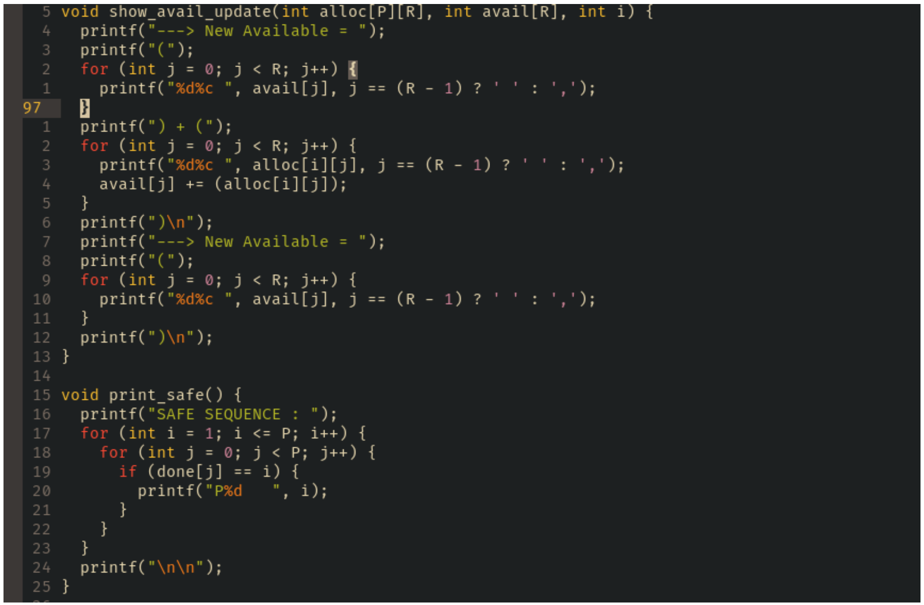
```c
void fill_need_matrix(int alloc[P][R], int max[P][R], int need[P][R]) {
  for (int i = 0; i < P; i++) {
    for (int j = 0; j < R; j++) {
      need[i][j] = max[i][j] - alloc[i][j];
    }
  }
}
```



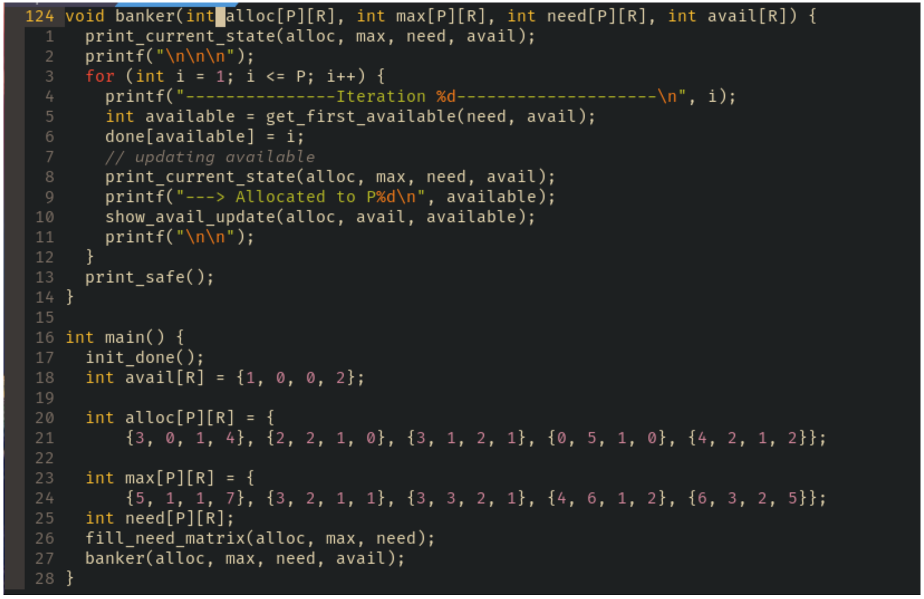
```c
void print_current_state(int alloc[P][R], int max[P][R], int need[P][R],
                         int avail[R]) {
  printf("Currently Available: ");
  for (int i = 0; i < R; i++) {
    printf("%-4d", avail[i]);
  }
  printf("\n\n");
  char fmt_str[100];
  // Decoration at top
  sprintf(fmt_str, "%%-8s|%%-%ds|%%-%ds|%%-%ds|%%s\n", R * 4, R * 4, R * 4);
  printf(fmt_str, "Name", "Allocation", "Max", "Need", "Order");

  // 2nd line
  printf("%-8s|", "");
  char c = 'A';
  for (int i = 0; i < R; i++)
    printf("%-4c", c++);
  printf("|");
  c = 'A';
  for (int i = 0; i < R; i++)
    printf("%-4c", c++);
  printf("|");
  c = 'A';
  for (int i = 0; i < R; i++)
    printf("%-4c", c++);
  printf("|\n");
  for (int i = 0; i <= (3 * R * 4 + 8 + 10); i++)
    printf("-");
  printf("\n");
```



```c

  // main content
  for (int i = 0; i < P; i++) {
    char str[10];
    sprintf(str, "P%d", i);
    printf("%-8s|", str);
    for (int j = 0; j < R; j++)
      printf("%-4d", alloc[i][j]);
    printf("|");
    for (int j = 0; j < R; j++)
      printf("%-4d", max[i][j]);
    printf("|");
    for (int j = 0; j < R; j++)
      printf("%-4d", need[i][j]);
    printf("|");
    printf("%-4d\n", done[i]);
  }
  printf("\n");
}
```

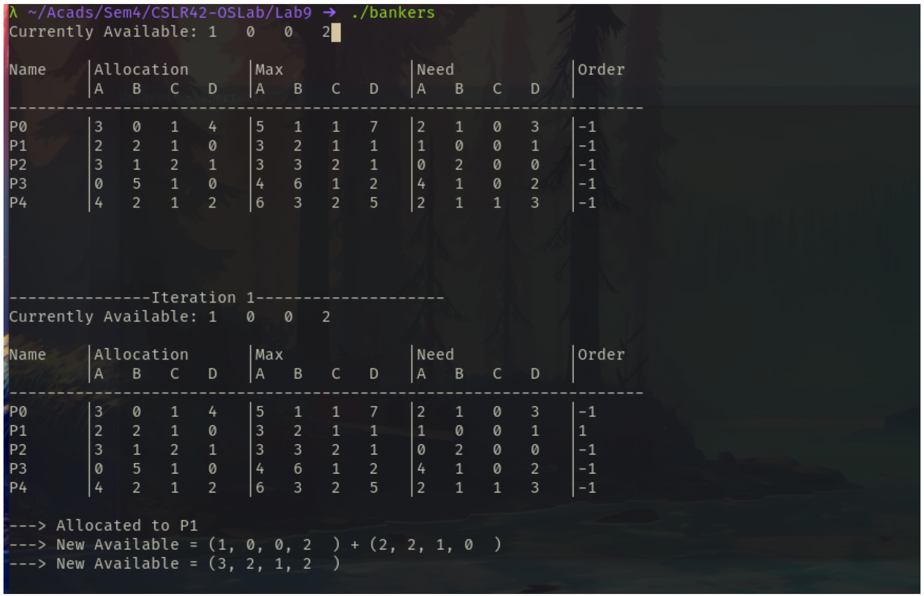


```c
void show_avail_update(int alloc[P][R], int avail[R], int i) {
  printf("---> New Available = ");
  printf("(");
  for (int j = 0; j < R; j++) {
    printf("%d%c ", avail[j], j == (R - 1) ? ' ' : ',');
  }
  printf(") + (");
  for (int j = 0; j < R; j++) {
    printf("%d%c ", alloc[i][j], j == (R - 1) ? ' ' : ',');
    avail[j] += (alloc[i][j]);
  }
  printf(")\n");
  printf("---> New Available = ");
  printf("(");
  for (int j = 0; j < R; j++) {
    printf("%d%c ", avail[j], j == (R - 1) ? ' ' : ',');
  }
  printf(")\n");
}

void print_safe() {
  printf("SAFE SEQUENCE : ");
  for (int i = 1; i <= P; i++) {
    for (int j = 0; j < P; j++) {
      if (done[j] == i) {
        printf("P%d   ", i);
      }
    }
  }
  printf("\n\n");
}
```

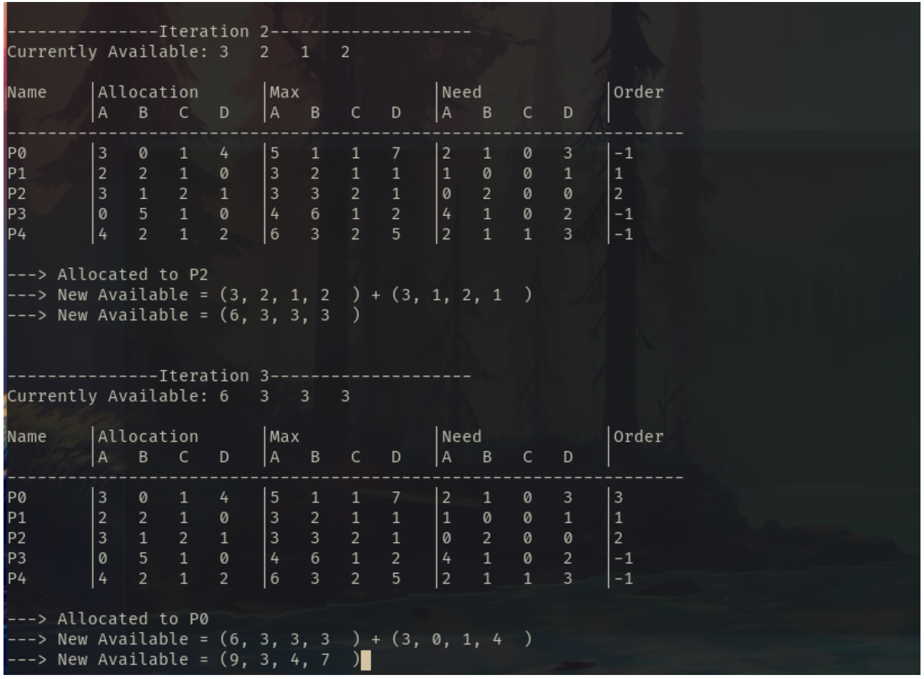


```c
void banker(int alloc[P][R], int max[P][R], int need[P][R], int avail[R]) {
  print_current_state(alloc, max, need, avail);
  printf("\n\n\n");
  for (int i = 1; i <= P; i++) {
    printf("---------------Iteration %d--------------------\n", i);
    int available = get_first_available(need, avail);
    done[available] = i;
    // updating available
    print_current_state(alloc, max, need, avail);
    printf("---> Allocated to P%d\n", available);
    show_avail_update(alloc, avail, available);
    printf("\n\n");
  }
  print_safe();
}

int main() {
  init_done();
  int avail[R] = {1, 0, 0, 2};

  int alloc[P][R] = {
      {3, 0, 1, 4}, {2, 2, 1, 0}, {3, 1, 2, 1}, {0, 5, 1, 0}, {4, 2, 1, 2}};

  int max[P][R] = {
      {5, 1, 1, 7}, {3, 2, 1, 1}, {3, 3, 2, 1}, {4, 6, 1, 2}, {6, 3, 2, 5}};
  int need[P][R];
  fill_need_matrix(alloc, max, need);
  banker(alloc, max, need, avail);
}
```

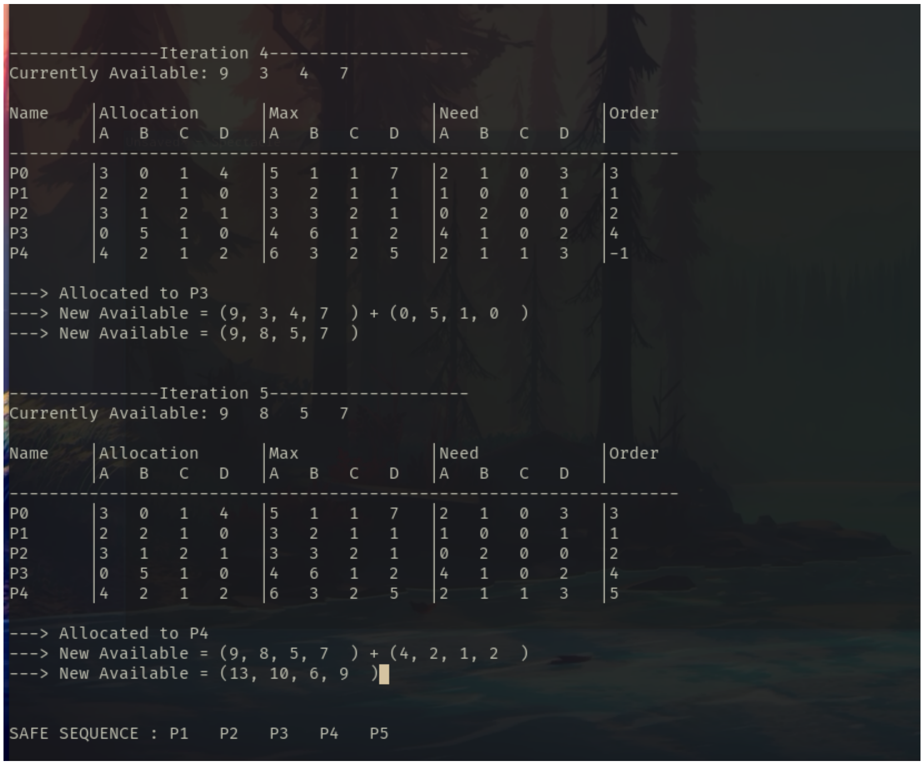
Output



```
λ ~/Acads/Sem4/CSLR42-OSLab/Lab9 ➔ ./bankers
Currently Available: 1   0   0   2

Name     |Allocation       |Max              |Need             |Order
         |A   B   C   D    |A   B   C   D    |A   B   C   D    |
-------------------------------------------------------------------
P0       |3   0   1   4    |5   1   1   7    |2   1   0   3    |-1
P1       |2   2   1   0    |3   2   1   1    |1   0   0   1    |-1
P2       |3   1   2   1    |3   3   2   1    |0   2   0   0    |-1
P3       |0   5   1   0    |4   6   1   2    |4   1   0   2    |-1
P4       |4   2   1   2    |6   3   2   5    |2   1   1   3    |-1



---------------Iteration 1--------------------
Currently Available: 1   0   0   2

Name     |Allocation       |Max              |Need             |Order
         |A   B   C   D    |A   B   C   D    |A   B   C   D    |
-------------------------------------------------------------------
P0       |3   0   1   4    |5   1   1   7    |2   1   0   3    |-1
P1       |2   2   1   0    |3   2   1   1    |1   0   0   1    |1
P2       |3   1   2   1    |3   3   2   1    |0   2   0   0    |-1
P3       |0   5   1   0    |4   6   1   2    |4   1   0   2    |-1
P4       |4   2   1   2    |6   3   2   5    |2   1   1   3    |-1

---> Allocated to P1
---> New Available = (1, 0, 0, 2   ) + (2, 2, 1, 0   )
---> New Available = (3, 2, 1, 2   )
```



```
---------------Iteration 2--------------------
Currently Available: 3   2   1   2

Name     |Allocation       |Max              |Need             |Order
         |A   B   C   D    |A   B   C   D    |A   B   C   D    |
-------------------------------------------------------------------
P0       |3   0   1   4    |5   1   1   7    |2   1   0   3    |-1
P1       |2   2   1   0    |3   2   1   1    |1   0   0   1    |1
P2       |3   1   2   1    |3   3   2   1    |0   2   0   0    |2
P3       |0   5   1   0    |4   6   1   2    |4   1   0   2    |-1
P4       |4   2   1   2    |6   3   2   5    |2   1   1   3    |-1

---> Allocated to P2
---> New Available = (3, 2, 1, 2   ) + (3, 1, 2, 1   )
---> New Available = (6, 3, 3, 3   )


---------------Iteration 3--------------------
Currently Available: 6   3   3   3

Name     |Allocation       |Max              |Need             |Order
         |A   B   C   D    |A   B   C   D    |A   B   C   D    |
-------------------------------------------------------------------
P0       |3   0   1   4    |5   1   1   7    |2   1   0   3    |3
P1       |2   2   1   0    |3   2   1   1    |1   0   0   1    |1
P2       |3   1   2   1    |3   3   2   1    |0   2   0   0    |2
P3       |0   5   1   0    |4   6   1   2    |4   1   0   2    |-1
P4       |4   2   1   2    |6   3   2   5    |2   1   1   3    |-1

---> Allocated to P0
---> New Available = (6, 3, 3, 3   ) + (3, 0, 1, 4   )
---> New Available = (9, 3, 4, 7   )
```



```
---------------Iteration 4--------------------
Currently Available: 9   3   4   7

Name     |Allocation       |Max              |Need             |Order
         |A   B   C   D    |A   B   C   D    |A   B   C   D    |
------------------------------------------------------------------
P0       |3   0   1   4    |5   1   1   7    |2   1   0   3    |3
P1       |2   2   1   0    |3   2   1   1    |1   0   0   1    |1
P2       |3   1   2   1    |3   3   2   1    |0   2   0   0    |2
P3       |0   5   1   0    |4   6   1   2    |4   1   0   2    |4
P4       |4   2   1   2    |6   3   2   5    |2   1   1   3    |-1

---> Allocated to P3
---> New Available = (9, 3, 4, 7  ) + (0, 5, 1, 0  )
---> New Available = (9, 8, 5, 7  )


---------------Iteration 5--------------------
Currently Available: 9   8   5   7

Name     |Allocation       |Max              |Need             |Order
         |A   B   C   D    |A   B   C   D    |A   B   C   D    |
------------------------------------------------------------------
P0       |3   0   1   4    |5   1   1   7    |2   1   0   3    |3
P1       |2   2   1   0    |3   2   1   1    |1   0   0   1    |1
P2       |3   1   2   1    |3   3   2   1    |0   2   0   0    |2
P3       |0   5   1   0    |4   6   1   2    |4   1   0   2    |4
P4       |4   2   1   2    |6   3   2   5    |2   1   1   3    |5

---> Allocated to P4
---> New Available = (9, 8, 5, 7  ) + (4, 2, 1, 2  )
---> New Available = (13, 10, 6, 9  )


SAFE SEQUENCE : P1   P2   P3   P4   P5
```