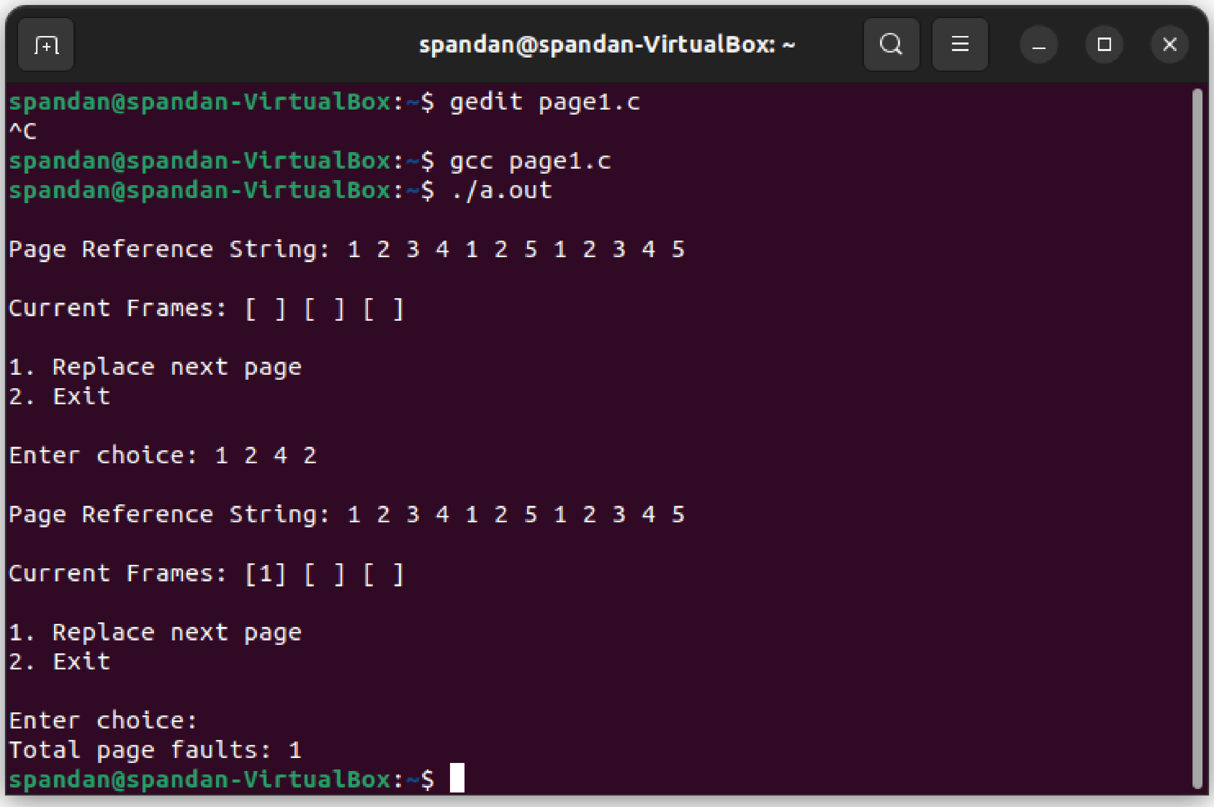
Name: Spandan Mukherjee
Registration Number: 21BCE1132
Subject: Operating Systems Lab
Topic: Page Replacement Algorithms

1)Develop page replacement algortihms (FIFO, Optimal, LRU) as a menu driven program and perform the replacements as per the menu choice selected by the user for the frame size -3. Assume your own frame reference string of size n and display the count of page fault

FIFO
CODE:
```
#include <stdio.h>

#define MAX_FRAMES 3

int main() {
   int frames[MAX_FRAMES], ref_string[] = {1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5};
   int ref_len = sizeof(ref_string) / sizeof(ref_string[0]);
   int faults = 0, curr_frame = 0, curr_page = 0, choice;

   // initialize frames to -1 (indicating an empty frame)
   for (int i = 0; i < MAX_FRAMES; i++) {
      frames[i] = -1;
   }

   do {
      printf("\nPage Reference String: ");
      for (int i = 0; i < ref_len; i++) {
         printf("%d ", ref_string[i]);
      }
      printf("\n\nCurrent Frames: ");
      for (int i = 0; i < MAX_FRAMES; i++) {
         if (frames[i] == -1) {
            printf("[ ] ");
         } else {
            printf("[%d] ", frames[i]);
         }
      }
      printf("\n\n1. Replace next page\n2. Exit\n\nEnter choice: ");
      scanf("%d", &choice);

      switch (choice) {
         case 1:
            if (curr_frame < MAX_FRAMES) {
               frames[curr_frame] = ref_string[curr_page];
               curr_frame++;
               curr_page++;
               faults++;
            } else {
               int found = 0;
               for (int i = 0; i < MAX_FRAMES; i++) {
                    if (frames[i] == ref_string[curr_page]) {
                        found = 1;
                        break;
                    }
                }
                if (!found) {
                    frames[curr_frame % MAX_FRAMES] = ref_string[curr_page];
                    curr_frame++;
                    faults++;
                }
                curr_page++;
            }
            break;
        case 2:
            break;
        default:
            printf("\nInvalid choice. Try again.\n");
            break;
        }
    } while (choice != 2);

    printf("\nTotal page faults: %d\n", faults);

    return 0;
}
```

OUTPUT:

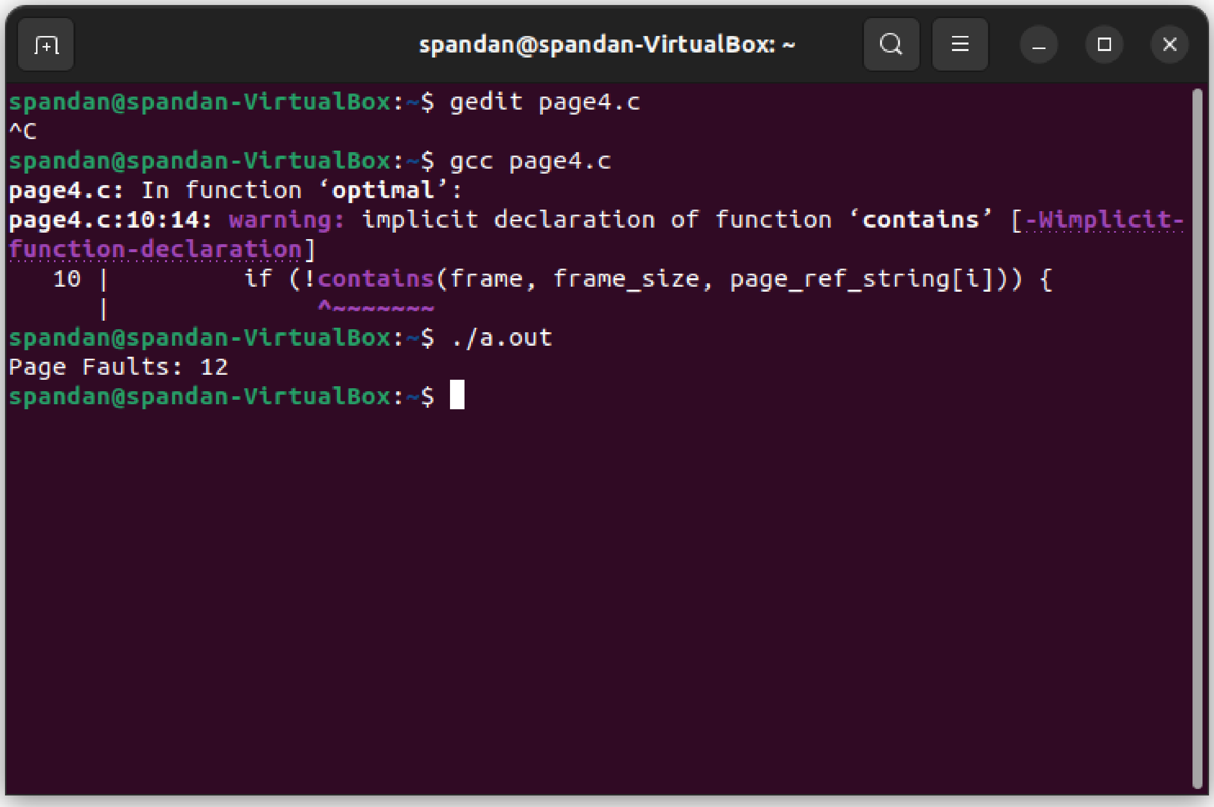


```
spandan@spandan-VirtualBox:~$ gedit page1.c
^C
spandan@spandan-VirtualBox:~$ gcc page1.c
spandan@spandan-VirtualBox:~$ ./a.out

Page Reference String: 1 2 3 4 1 2 5 1 2 3 4 5

Current Frames: [ ] [ ] [ ]

1. Replace next page
2. Exit

Enter choice: 1 2 4 2

Page Reference String: 1 2 3 4 1 2 5 1 2 3 4 5

Current Frames: [1] [ ] [ ]

1. Replace next page
2. Exit

Enter choice: 
Total page faults: 1
spandan@spandan-VirtualBox:~$ 
```

Optimal

CODE:
```
#include <stdio.h>

int optimal(int page_ref_string[], int n, int frame_size) {
    int page_faults = 0;
    int frame[frame_size];
    int next_ref[frame_size];
    int i, j, k, max;

    for (i = 0; i < n; i++) {
        if (!contains(frame, frame_size, page_ref_string[i])) {
            page_faults++;

            if (i < frame_size) {
                frame[i] = page_ref_string[i];
            } else {
                for (j = 0; j < frame_size; j++) {
                    if (!contains(&page_ref_string[i], n-i, frame[j])) {
                        next_ref[j] = n;
                    } else {
                        for (k = i; k < n; k++) {
                            if (page_ref_string[k] == frame[j]) {
                                next_ref[j] = k;
                                break;
                            }
                        }
                    }
                }

                max = 0;
                for (j = 1; j < frame_size; j++) {
                    if (next_ref[j] > next_ref[max]) {
                        max = j;
                    }
                }

                frame[max] = page_ref_string[i];
            }
        }
    }

    return page_faults;
}

int contains(int arr[], int n, int x) {
    int i;

    for (i = 0; i < n; i++) {
        if (arr[i] == x) {
            return 1;
        }
    }

    return 0;
}

int main() {
    int page_ref_string[] = {1, 2, 3, 4, 1, 5, 6, 2, 1, 3, 7, 6, 3, 2, 1, 2, 3, 6, 7, 3, 1};
    int n = sizeof(page_ref_string) / sizeof(int);
    int frame_size = 3;

    printf("Page Faults: %d\n", optimal(page_ref_string, n, frame_size));

    return 0;
}
```

OUTPUT:

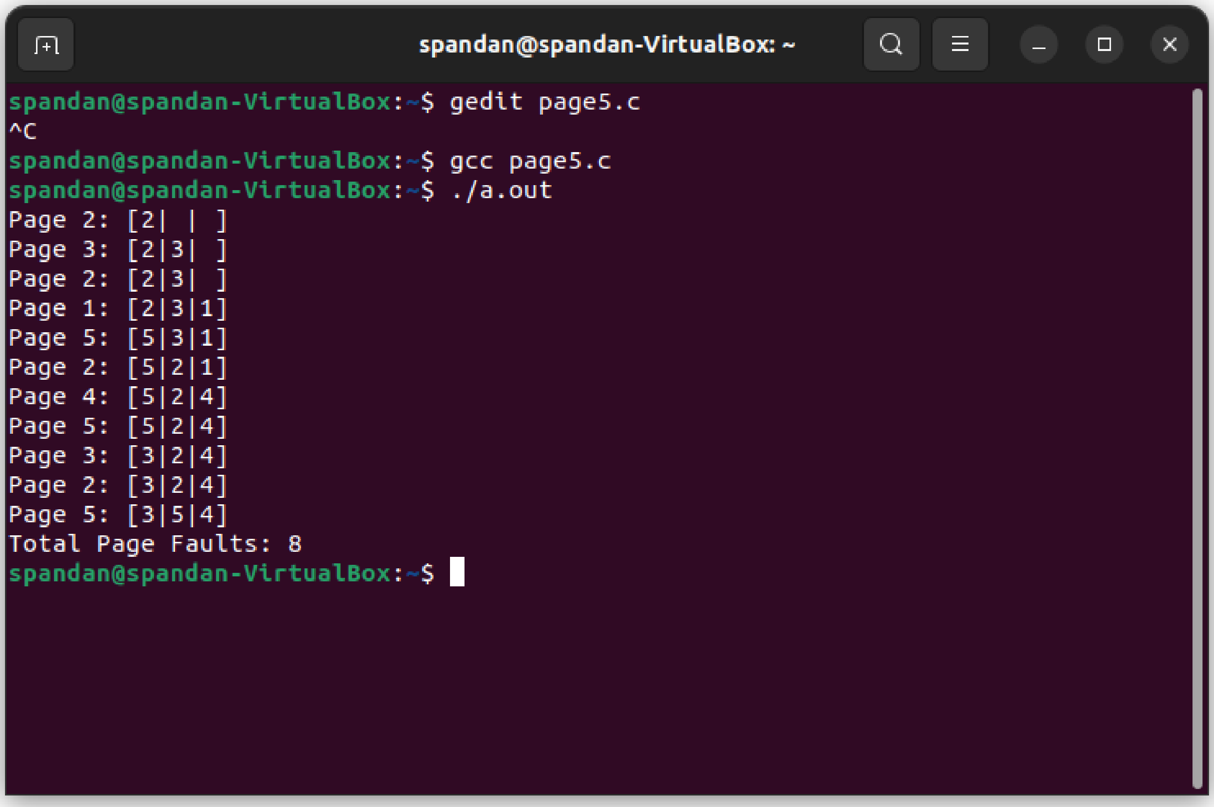


```
spandan@spandan-VirtualBox:~$ gedit page4.c
^C
spandan@spandan-VirtualBox:~$ gcc page4.c
page4.c: In function ‘optimal’:
page4.c:10:14: warning: implicit declaration of function ‘contains’ [-Wimplicit-function-declaration]
   10 |         if (!contains(frame, frame_size, page_ref_string[i])) {
      |              ^~~~~~~~
spandan@spandan-VirtualBox:~$ ./a.out
Page Faults: 12
spandan@spandan-VirtualBox:~$
```

LRU:

CODE:
```c
#include <stdio.h>

#define FRAME_SIZE 3

int main() {
    int frame[FRAME_SIZE];
    int page_faults = 0;
    int index = 0;
    int reference_string[] = {2, 3, 2, 1, 5, 2, 4, 5, 3, 2, 5};

    // Initialize frame to -1 to indicate an empty slot
    for (int i = 0; i < FRAME_SIZE; i++) {
        frame[i] = -1;
    }

    // Iterate through the reference string
    for (int i = 0; i < sizeof(reference_string)/sizeof(int); i++) {
        int page_number = reference_string[i];
        int page_found = 0;

        // Check if page is already in frame
        for (int j = 0; j < FRAME_SIZE; j++) {
            if (frame[j] == page_number) {
                page_found = 1;
                break;
            }
        }

        // If page is not in frame, replace LRU page
        if (!page_found) {
            frame[index] = page_number;
            index = (index + 1) % FRAME_SIZE;
            page_faults++;
        }

        // Display current state of frame
        printf("Page %d: [", page_number);
        for (int j = 0; j < FRAME_SIZE; j++) {
            if (frame[j] == -1) {
                printf(" ");
            } else {
                printf("%d", frame[j]);
            }
            if (j < FRAME_SIZE - 1) {
                printf("|");
            }
        }
        printf("]\n");
    }
    // Display total number of page faults
    printf("Total Page Faults: %d\n", page_faults);

    return 0;
}
```

OUTPUT:

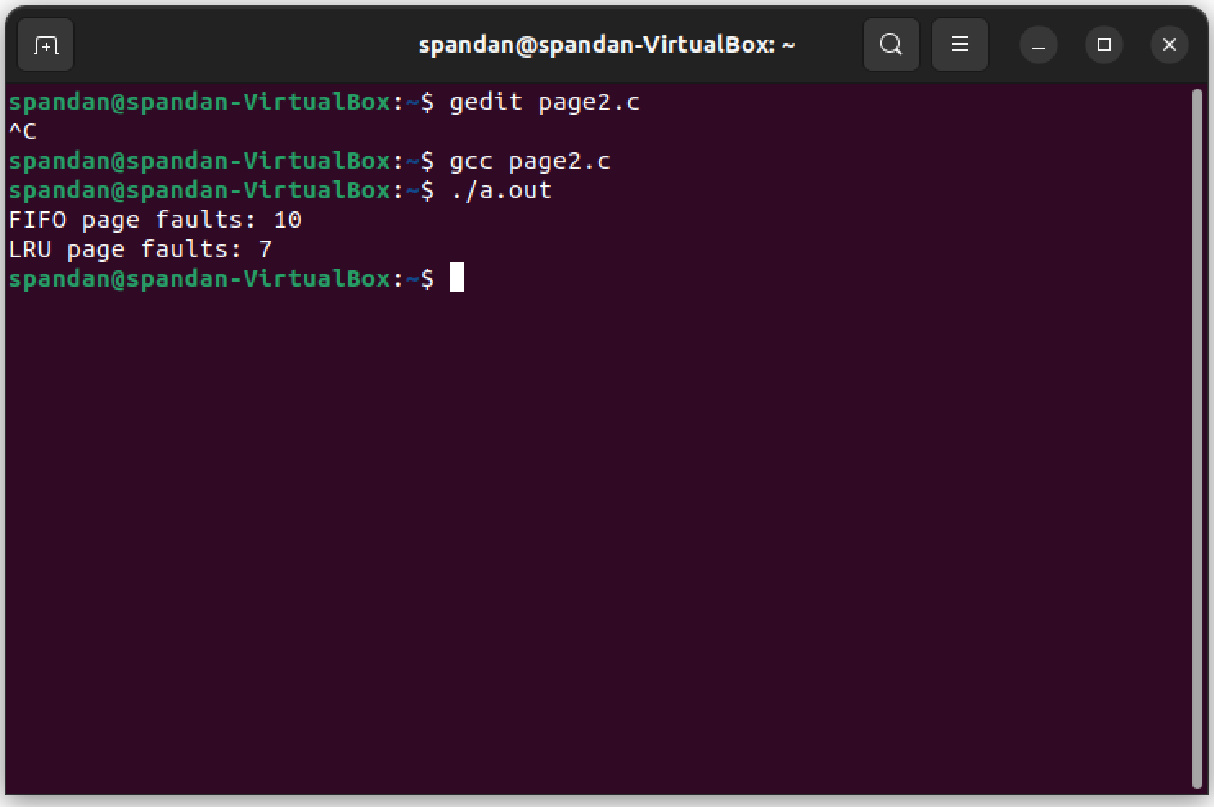


```
spandan@spandan-VirtualBox:~$ gedit page5.c
^C
spandan@spandan-VirtualBox:~$ gcc page5.c
spandan@spandan-VirtualBox:~$ ./a.out
Page 2: [2| | ]
Page 3: [2|3| ]
Page 2: [2|3| ]
Page 1: [2|3|1]
Page 5: [5|3|1]
Page 2: [5|2|1]
Page 4: [5|2|4]
Page 5: [5|2|4]
Page 3: [3|2|4]
Page 2: [3|2|4]
Page 5: [3|5|4]
Total Page Faults: 8
spandan@spandan-VirtualBox:~$ 
```

2) Develop page replacement algorithms FIFO, LRU using two threads and let them return the page faults of those algorithms with the frame size 4. Assume your own frame reference string of size n.

CODE:
```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

#define FRAME_SIZE 4

int reference_string[] = {1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5}; // example reference string
int n = sizeof(reference_string) / sizeof(reference_string[0]); // length of reference string

int page_faults_fifo = 0; // counter for page faults of FIFO algorithm
int page_faults_lru = 0; // counter for page faults of LRU algorithm

int frame_fifo[FRAME_SIZE]; // frame for FIFO algorithm
int frame_lru[FRAME_SIZE]; // frame for LRU algorithm
int age_lru[FRAME_SIZE]; // age of each page in the LRU frame

pthread_mutex_t mutex_fifo; // mutex for FIFO algorithm
pthread_mutex_t mutex_lru; // mutex for LRU algorithm

void* fifo(void* arg) {
   int i, j, k;
   for (i = 0; i < n; i++) {
      int page = reference_string[i];
      int hit = 0;
      pthread_mutex_lock(&mutex_fifo);
      for (j = 0; j < FRAME_SIZE; j++) {
         if (frame_fifo[j] == page) {
            hit = 1;
            break;
         }
      }
      if (!hit) {
         page_faults_fifo++;
         for (j = 0; j < FRAME_SIZE - 1; j++) {
            frame_fifo[j] = frame_fifo[j+1];
         }
         frame_fifo[FRAME_SIZE - 1] = page;
      }
      pthread_mutex_unlock(&mutex_fifo);
   }
   return NULL;
}

void* lru(void* arg) {
   int i, j, k;
   for (i = 0; i < n; i++) {
        int page = reference_string[i];
        int hit = 0;
        pthread_mutex_lock(&mutex_lru);
        for (j = 0; j < FRAME_SIZE; j++) {
            if (frame_lru[j] == page) {
                hit = 1;
                age_lru[j] = 0;
                break;
            }
        }
        if (!hit) {
            page_faults_lru++;
            int max_age = 0;
            int max_age_index = 0;
            for (j = 0; j < FRAME_SIZE; j++) {
                age_lru[j]++;
                if (age_lru[j] > max_age) {
                    max_age = age_lru[j];
                    max_age_index = j;
                }
            }
            frame_lru[max_age_index] = page;
            age_lru[max_age_index] = 0;
        }
        pthread_mutex_unlock(&mutex_lru);
    }
    return NULL;
}

int main() {
    pthread_t thread_fifo, thread_lru;
    pthread_mutex_init(&mutex_fifo, NULL);
    pthread_mutex_init(&mutex_lru, NULL);
    pthread_create(&thread_fifo, NULL, fifo, NULL);
    pthread_create(&thread_lru, NULL, lru, NULL);
    pthread_join(thread_fifo, NULL);
    pthread_join(thread_lru, NULL);
    pthread_mutex_destroy(&mutex_fifo);
    pthread_mutex_destroy(&mutex_lru);
    printf("FIFO page faults: %d\n", page_faults_fifo);
    printf("LRU page faults: %d\n", page_faults_lru);
    return 0;
}
```

OUTPUT:

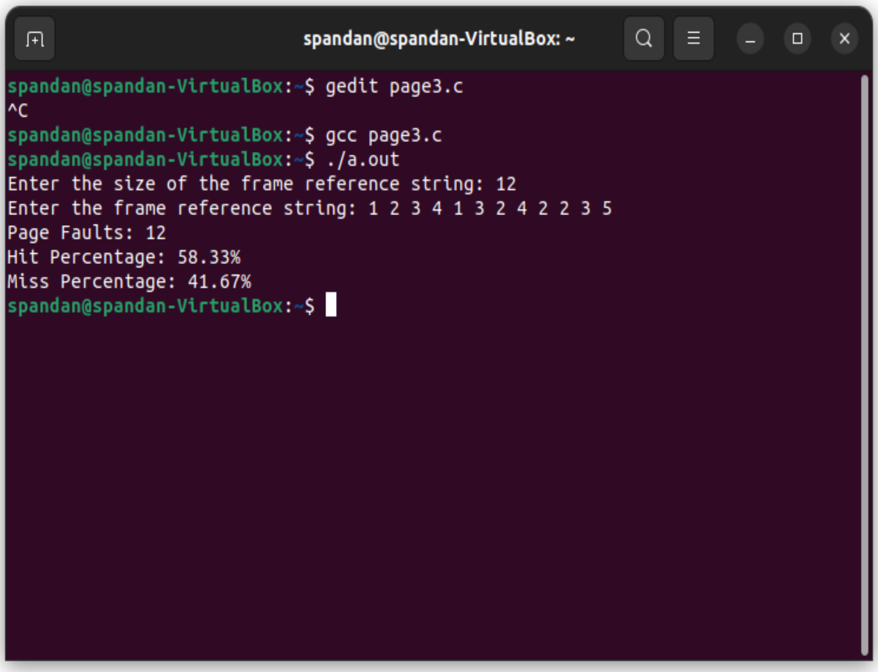


```
spandan@spandan-VirtualBox:~$ gedit page2.c
^C
spandan@spandan-VirtualBox:~$ gcc page2.c
spandan@spandan-VirtualBox:~$ ./a.out
FIFO page faults: 10
LRU page faults: 7
spandan@spandan-VirtualBox:~$ 
```

3) Develop second chance page replacement algorithmwith the frame size 4. Assume your own frame reference string of size n. Display the page fault and hit percentage.

CODE:

```
#include <stdio.h>

#define FRAME_SIZE 4

int frames[FRAME_SIZE];
int second_chance[FRAME_SIZE];

int main() {
   int n, page_faults = 0, hits = 0;
   printf("Enter the size of the frame reference string: ");
   scanf("%d", &n);
   int reference_string[n];
   printf("Enter the frame reference string: ");
   for (int i = 0; i < n; i++) {
      scanf("%d", &reference_string[i]);
   }
   for (int i = 0; i < FRAME_SIZE; i++) {
      frames[i] = -1;
      second_chance[i] = 0;
   }
   int index = 0;
   for (int i = 0; i < n; i++) {
        int page = reference_string[i];
        int frame_index = -1;
        for (int j = 0; j < FRAME_SIZE; j++) {
            if (frames[j] == page) {
                frame_index = j;
                hits++;
                second_chance[frame_index] = 1;
                break;
            }
            if (frames[j] == -1) {
                frame_index = j;
                break;
            }
        }
        if (frame_index == -1) {
            while (1) {
                if (second_chance[index] == 0) {
                    frame_index = index;
                    break;
                }
                second_chance[index] = 0;
                index = (index + 1) % FRAME_SIZE;
            }
        }
        frames[frame_index] = page;
        second_chance[frame_index] = 1;
        page_faults++;
        index = (index + 1) % FRAME_SIZE;
    }
    printf("Page Faults: %d\n", page_faults);
    printf("Hit Percentage: %.2f%%\n", (float)hits / n * 100);
    printf("Miss Percentage: %.2f%%\n", (float)(page_faults - hits) / n * 100);
    return 0;
}
```
OUTPUT:



```
spandan@spandan-VirtualBox:~$ gedit page3.c
^C
spandan@spandan-VirtualBox:~$ gcc page3.c
spandan@spandan-VirtualBox:~$ ./a.out
Enter the size of the frame reference string: 12
Enter the frame reference string: 1 2 3 4 1 3 2 4 2 2 3 5
Page Faults: 12
Hit Percentage: 58.33%
Miss Percentage: 41.67%
spandan@spandan-VirtualBox:~$
```