//sequential file allocation

#include <stdio.h>

#include <stdlib.h>

#define TOTAL\_BLOCKS 16

#define BLOCK\_SIZE 1024

#define INODE\_BLOCKS 8

#define MIN\_FILE\_SIZE 1024

int areBlocksFree(int startBlock, int numBlocks, int \*allocationMap)

{

for (int i = startBlock; i < startBlock + numBlocks; ++i)

{

if (allocationMap[i] == 1)

{

return 0;

}

}

return 1;

}

void allocateBlocks(int startBlock, int numBlocks, int \*allocationMap)

{

for (int i = startBlock; i < startBlock + numBlocks; ++i)

{

allocationMap[i] = 1;

}

}

int main()

{

int allocationMap[TOTAL\_BLOCKS] = {0};

int fileSize;

printf("Enter the file size (minimum 1 KB): ");

scanf("%d", &fileSize);

if (fileSize < MIN\_FILE\_SIZE)

{

printf("Error: File size should be at least 1 KB.\n");

exit(EXIT\_FAILURE);

}

int numBlocksRequired = (fileSize + BLOCK\_SIZE - 1) / BLOCK\_SIZE;

int startBlock = -1;

for (int i = INODE\_BLOCKS; i < TOTAL\_BLOCKS - numBlocksRequired + 1; ++i)

{

if (areBlocksFree(i, numBlocksRequired, allocationMap))

{

startBlock = i;

allocateBlocks(startBlock, numBlocksRequired, allocationMap);

break;

}

}

if (startBlock == -1)

{

printf("Error: Not enough contiguous free blocks available.\n");

exit(EXIT\_FAILURE);

}

printf("File allocated to blocks: ");

for (int i = startBlock; i < startBlock + numBlocksRequired; ++i) {

printf("%d ", i);

}

printf("\n");

return 0;

}

Output:

