/\*

GOPIKRISHNA V

S3 CSE A

52

\*/

#include<stdio.h>

#include<stdlib.h>

int g = 0, k = 0;

struct free

{

int tag;

int size;

struct free\* next;

}\*free\_head = NULL,\*prev\_free = NULL;

struct alloc

{

int block\_id;

int tag;

int size;

struct alloc\* next;

}\*alloc\_head = NULL,\*prev\_alloc = NULL;

void create\_free(int c)

{

struct free\* p = (struct free\*)malloc(sizeof(struct free));

p->size = c;

p->tag = g;

p->next = NULL;

if (free\_head == NULL)

free\_head = p;

else

prev\_free->next = p;

prev\_free = p;

g++;

}

void print\_free()

{

struct free\* p = free\_head;

printf("Tag\tSize\n");

while (p != NULL)

{

printf("%d <> %d\n",p->tag,p->size);

p = p->next;

}

}

void print\_alloc()

{

struct alloc\* p = alloc\_head;

printf("Tag\tBlock ID\tSize\n");

while (p != NULL)

{

printf("%d\t%d\t\t%d\n",p->tag,p->block\_id,p->size);

p = p->next;

}

}

void create\_alloc(int c)

{

struct alloc\* q = (struct alloc\*)malloc(sizeof(struct alloc));

q->size = c;

q->tag = k;

q->next = NULL;

struct free\* p = free\_head;

struct free\* r = (struct free\*)malloc(sizeof(struct free));

r->size = 99999;

while (p != NULL)

{

if (q->size <= p->size)

{

if (p->size < r->size)

r = p;

}

p = p->next;

}

if (r->size != 99999)

{

q->block\_id = r->tag;

r->size = q->size;

if (alloc\_head == NULL)

alloc\_head = q;

else

{

prev\_alloc = alloc\_head;

while (prev\_alloc->next != NULL)

prev\_alloc = prev\_alloc->next;

prev\_alloc->next = q;

}

k++;

}

else

printf("Block with size %d can't be allocated\n",c);

}

void delete\_alloc(int t)

{

struct alloc \*p = alloc\_head, \*q = NULL;

while (p != NULL)

{

if (p->tag == t)

break;

q = p;

p = p->next;

}

if (p == NULL)

printf("Tag ID doesn't exist\n");

else

{

if (p == alloc\_head)

alloc\_head = alloc\_head->next;

else

q->next = p->next;

}

struct free\* temp = free\_head;

while (temp != NULL)

{

if (temp->tag == p->block\_id)

{

temp->size += p->size;

break;

}

temp = temp->next;

}

}

void main()

{

int lim;

printf("Enter number of Size Blocks = ");

scanf("%d",&lim);

int blockSize[lim];

for (int i = 0; i < lim; i++)

{

printf("Size Block %d >> ",i+1);

scanf("%d",&blockSize[i]);

}

printf("Enter number of Process Blocks = ");

scanf("%d",&lim);

int processSize[lim];

for (int i = 0; i < lim; i++)

{

printf("Process Block %d >> ",i+1);

scanf("%d",&processSize[i]);

}

int m = sizeof(blockSize)/ sizeof(blockSize[0]);

int n = sizeof(processSize)/ sizeof(processSize[0]);

for (int i = 0; i < m; i++)

create\_free(blockSize[i]);

for (int i = 0; i < n; i++)

create\_alloc(processSize[i]);

print\_alloc();

delete\_alloc(1);

create\_alloc(426);

printf("After deleting block with Tag ID 1.\n");

print\_alloc();

}

**OUTPUT**

****