Instituto Tecnológico de Costa Rica

Escuela de Computación

Curso: Sistemas Operativos

Tarea 6

El módulo *mem*, agregado a *libtask*, cuenta con una serie de funciones para administración de memoria similares a las proporcionadas por el lenguaje C. Las funciones típicas para el manejo de memoria son: *malloc*, *calloc*, *free* y *realloc*.

El siguiente ejemplo muestra un programa que crea varias tareas donde cada una genera matrices de tamaños aleatorios y las llena también con valores aleatorios. Al final cada tarea invoca una función que imprime la matriz generada. Note como se asigna la memoria para la matriz y como se libera.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include "task.h"
enum {STACK = 32768};
void printMatrix(void* _matrix,int cols,int rows) {
    int (*matrix)[cols] = _matrix;
    printf("rows: %d, cols: %d\n",rows,cols);
    for (int i=0; i<cols; i++) {
        for (int j=0; j<rows; j++) {
            printf("%d",matrix[i][j]);
        }
        printf("\n");
    }
    printf("\n");
}
void myTask(void *arg) {
    int rowCount = rand()%19+1;
    int colCount = rand()%19+1;
    int (*matrix)[colCount];
    matrix = malloc(sizeof(int)*rowCount*colCount);
    for (int i=0; i<colCount; i++) {
        for (int j=0; j<rowCount; j++) {
            matrix[i][j] = rand()%10;
        }
        taskyield();
    }
    printMatrix(matrix,colCount,rowCount);
```

```
    free(matrix);
    taskexit(0);
}
void taskmain(int argc, char **argv) {
  for (int i=0; i < 5; i++) {
    taskcreate(myTask,(void*)i,STACK);
  }
}
```

Ejercicios

1. Modifique el módulo *mem* para incluir una nueva función *void meminfo()* que despliegue en pantalla la información de todos los bloques de memoria asignados. Debe imprimir la dirección en donde empieza y el tamaño.

2. Modifique la rutina *malloc* para que guarde el identificador de la tarea que realiza la solicitud de memoria. Modifique también la función *meminfo* del ejercicio anterior para que imprima el identificador de la tarea asociada al bloque de memoria.

3. Modifique el código de la rutina *void taskexit(int status)* para que libere todos los bloques de memoria asociados a la tarea que termina y así no requerir de hacer la llamada a *free* de forma explícita.