



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Adrian Ulises Mercado Martinez

Profesor:

Estructura de Datos y Algoritmos I.

Asignatura:

13

Grupo:

12

No de Práctica(s):

Nicolás López Daniela

Integrante(s):

*No. de Equipo de
cómputo empleado:*

-

36

No. de Lista o Brigada:

2020-2

Semestre:

07-06-2020

Fecha de entrega:

Observaciones:

CALIFICACIÓN: _____

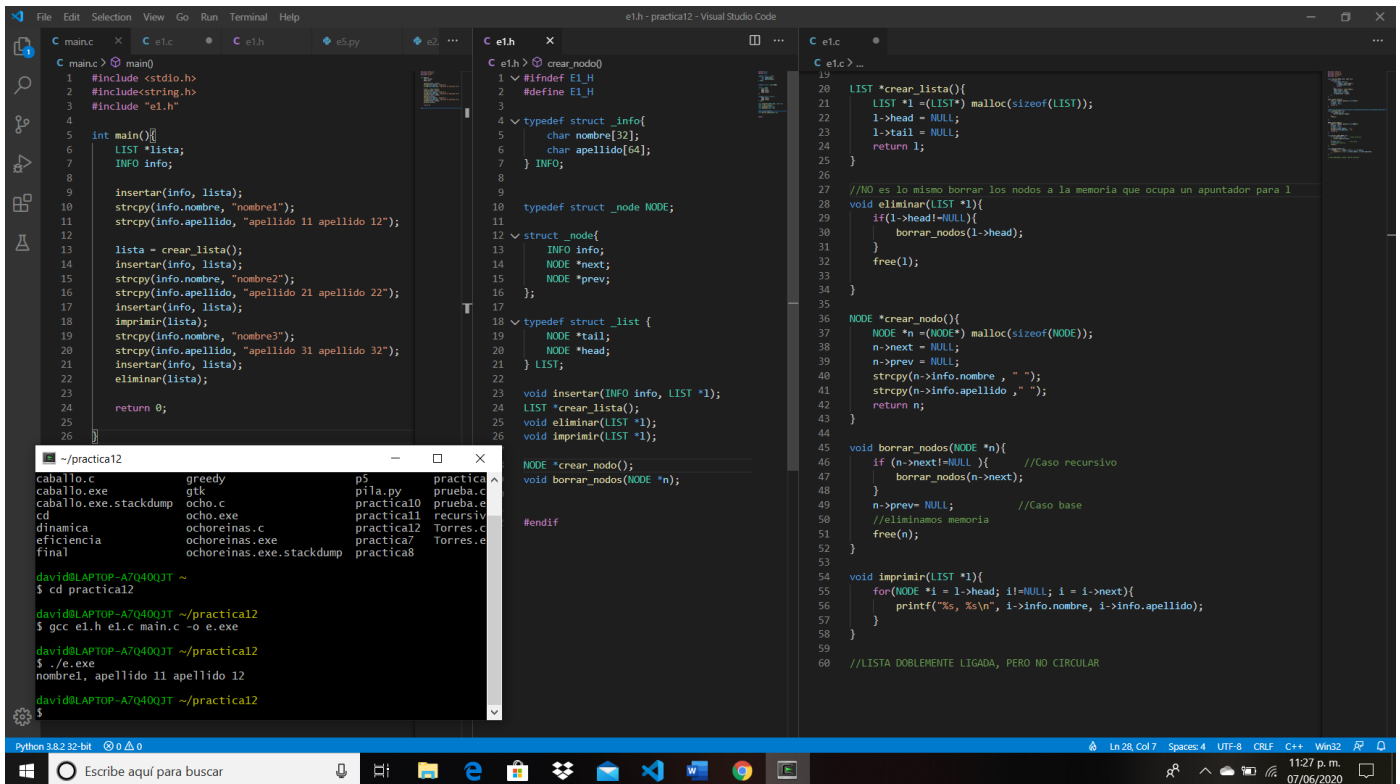
❖ INTRODUCCION.

En esta practica realizamos 3 programas en los cuales creamos por medio de diferentes librerias y funciones algoritmos de distintos tipos, además de que por medio de la creacion de graficas pudimos ver la eficiencia de algunos.

❖ DESARROLLO (CON EJERCICIOS)

■ PROGRAMA 1.

En este primer programa creamos el punto c, h y el main de un programa, se trata de una lista doblemente ligada circular pero que tiene casos recursivos dentro de si mismo.



```
1 #include <stdio.h>
2 #include <string.h>
3 #include "e1.h"
4
5 int main()
6 {
7     LIST *lista;
8     INFO info;
9
10    insertar(info, lista);
11    strcpy(info.nombre, "nombre1");
12    strcpy(info.apellido, "apellido 11 apellido 12");
13
14    lista = crear_lista();
15    insertar(info, lista);
16    strcpy(info.nombre, "nombre2");
17    strcpy(info.apellido, "apellido 21 apellido 22");
18    insertar(info, lista);
19    imprimir(lista);
20    strcpy(info.nombre, "nombre3");
21    strcpy(info.apellido, "apellido 31 apellido 32");
22    insertar(info, lista);
23    eliminar(lista);
24
25    return 0;
26 }
```

```
1 #ifndef E1_H
2 #define E1_H
3
4 typedef struct _info{
5     char nombre[32];
6     char apellido[64];
7 } INFO;
8
9
10 typedef struct _node NODE;
11
12 struct _node{
13     INFO info;
14     NODE *next;
15     NODE *prev;
16 };
17
18 typedef struct _list {
19     NODE *tail;
20     NODE *head;
21 } LIST;
22
23 void insertar(INFO info, LIST *l);
24 LIST *crear_lista();
25 void eliminar(LIST *l);
26 void imprimir(LIST *l);
27
28 NODE *crear_nodo();
29 void borrar_nodos(NODE *n);
30
31 #endif
```

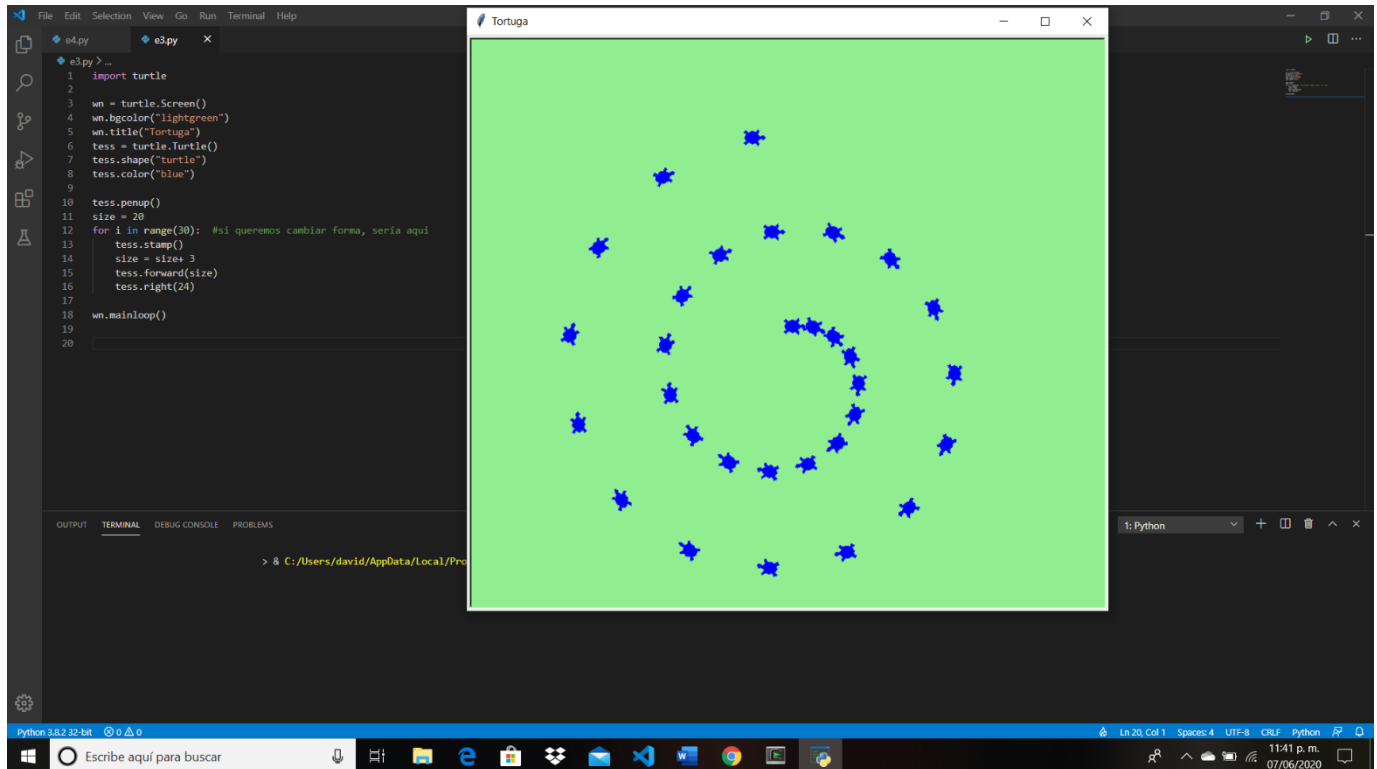
```
19
20 LIST *crear_lista(){
21     LIST *l = (LIST*) malloc(sizeof(LIST));
22     l->head = NULL;
23     l->tail = NULL;
24     return l;
25 }
26
27 //NO es lo mismo borrar los nodos a la memoria que ocupa un apuntador para l
28 void eliminar(LIST *l){
29     if(l->head!=NULL){
30         borrar_nodos(l->head);
31         free(l);
32     }
33 }
34
35
36 NODE *crear_nodo(){
37     NODE *n = (NODE*) malloc(sizeof(NODE));
38     n->next = NULL;
39     n->prev = NULL;
40     strcpy(n->info.nombre, " ");
41     strcpy(n->info.apellido, " ");
42     return n;
43 }
44
45 void borrar_nodos(NODE *n){
46     if (n->next!=NULL ){ //Caso recursivo
47         borrar_nodos(n->next);
48     }
49     n->prev= NULL; //Caso base
50     //eliminamos memoria
51     free(n);
52 }
53
54 void imprimir(LIST *l){
55     for(NODE *i = l->head; i!=NULL; i = i->next){
56         printf("%s, %s\n", i->info.nombre, i->info.apellido);
57     }
58 }
59
60 //LISTA DOBLEMENTE LIGADA, PERO NO CIRCULAR
```

```
~/practica12
caballo.c      greedy      p5          practica8
caballo.exe    gtk         pila.py     prueba.c
caballo.exe.stackdump  ocho.c      practica10  prueba.c
cd             ocho.exe    practica11  recursiv
dinamica       ochoreinas.c practica12  Torres.c
eficiencia     ochoreinas.exe stackdump   practica8
final          ochoreinas.exe.stackdump  practica8
```

```
david@LAPTOP-A7Q40QJ7 ~
$ cd practica12
david@LAPTOP-A7Q40QJ7 ~/practica12
$ gcc e1.h e1.c main.c -o e.exe
david@LAPTOP-A7Q40QJ7 ~/practica12
$ ./e.exe
nombre1, apellido 11 apellido 12
david@LAPTOP-A7Q40QJ7 ~/practica12
$
```

- PROGRAMA 2.

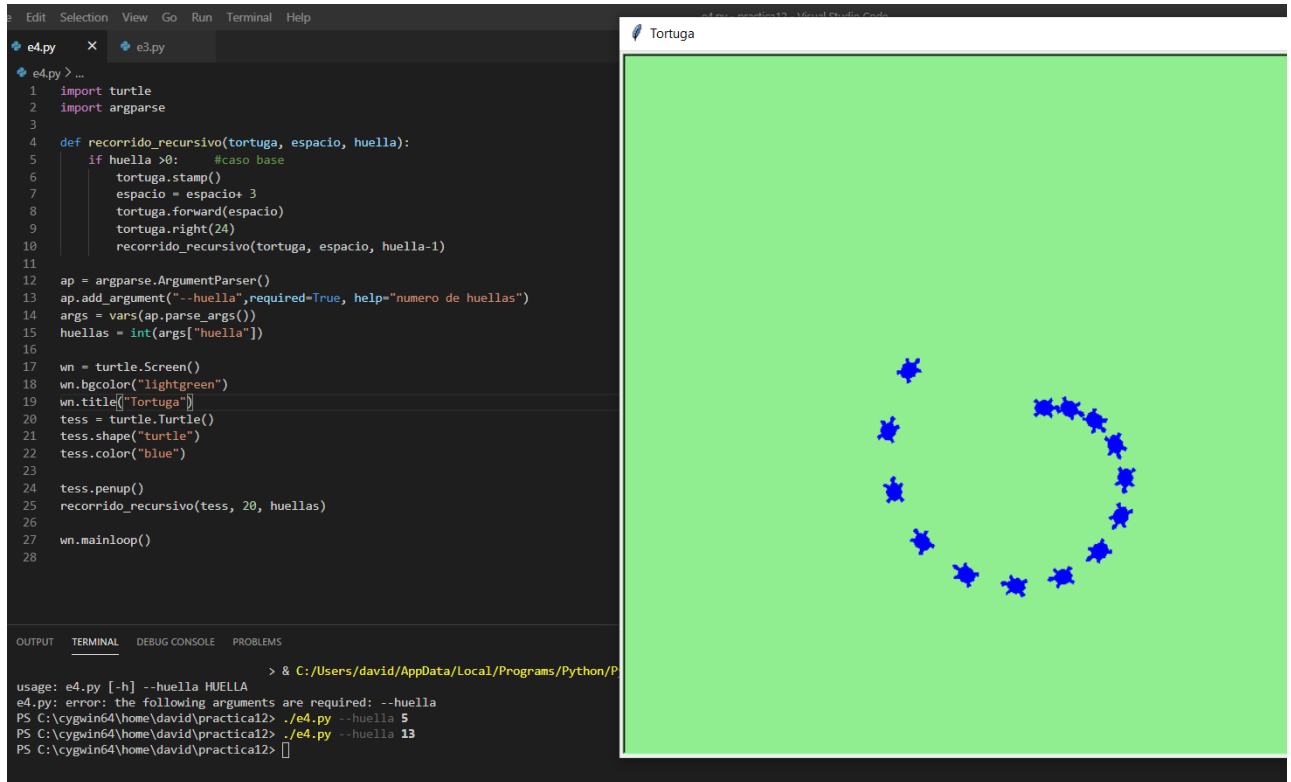
En este programa importamos una libreria que se llama Tortuga. La cual se encarga de graficar siluetas de tortugas en nuestro programa, aqui le dimos un rango a las impresiones y por medio de un ciclo for se iban estampando nuestras tortugas, ya que al entrar se les daba cierto espacio y se repetía sucesivamente, asi se formaba nuestro algoritmo recursivo



■ PROGRAMA 3.

Para este ultimo programa volvimos a utilizar a turtle, por medio de este programa haciamos el recorrido de la Tortuga recursive, nuevamente, entramos a un ciclo, pero en este caso es un if, el cual al entrar estampa las tortugas.

Pero la diferencia con el anterior programa es que a este le podemos dar el número de ejecuciones que deseemos, debido a las huellasque queramos estampar.



The screenshot shows a Python IDE with a file named `e4.py` open. The code defines a recursive function `recorrido_recursivo` that uses the `turtle` module to draw a spiral pattern of blue stars on a light green background. The function takes three arguments: `tortuga` (the turtle object), `espacio` (the distance between stars), and `huella` (the number of stars to draw). The function calls itself recursively until `huella` reaches 0. The main part of the program sets up the turtle screen, creates a turtle object named `tess`, and calls the recursive function with `tess`, `20`, and the number of stars (passed as a command-line argument). The terminal window shows the command `python e4.py --huella 13` being executed, and the output shows the program running successfully.

```
1 import turtle
2 import argparse
3
4 def recorrido_recursivo(tortuga, espacio, huella):
5     if huella > 0: #caso base
6         tortuga.stamp()
7         espacio = espacio + 3
8         tortuga.forward(espacio)
9         tortuga.right(24)
10        recorrido_recursivo(tortuga, espacio, huella-1)
11
12 ap = argparse.ArgumentParser()
13 ap.add_argument("--huella", required=True, help="numero de huellas")
14 args = vars(ap.parse_args())
15 huellas = int(args["huella"])
16
17 wn = turtle.Screen()
18 wn.bgcolor("lightgreen")
19 wn.title("Tortuga")
20 tess = turtle.Turtle()
21 tess.shape("turtle")
22 tess.color("blue")
23
24 tess.penup()
25 recorrido_recursivo(tess, 20, huellas)
26
27 wn.mainloop()
28
```

usage: e4.py [-h] --huella HUELLA
e4.py: error: the following arguments are required: --huella
PS C:\cygwin64\home\david\practica12> ./e4.py --huella 5
PS C:\cygwin64\home\david\practica12> ./e4.py --huella 13
PS C:\cygwin64\home\david\practica12> █

CONCLUSIONES

-Esta práctica es entretenida ya que utilizamos graficos dinamicos que ayudaron a comprender en los programas los onceptos de recursividad, tenemos que tener los antecedentes de python completos para poder realizarla, además de los conocimientos en C para las estructuras de datos, me parece que fue algo en donde tuvimos que usar los conceptos de todo el curso.