Consignas parcial

1)
```
FILE * binario;
FILE * texto;
sinonimos_t s;

binario = fopen ("sinonimos.dat", "rb");
if (binario == NULL){
        printf("Error /n");
        return -1;
}
 if (fread(&s, sizeof(sinonimos_t), 1, binario)!= 1){
        printf("Error"); fclose(binario);
        return -1;
 }
 fclose(binario);
 texto = fopen("sinonimos.txt", "w");
 if (texto == NULL){
        printf ("Error /n");
        return -1;
 }
  fprintf(texto, "%s", s.palabra);
  for (int i=0; i< s.cant_sinonimos; i++){
        fprintf (texto, " | %s", s.sinonimos[i]);
   }
  fprintf("/n");
  return 0;
}
```

HOJA N°

FECHA

2) a) Depende donde se declare
no lo voy a hacer porque es modulo 1

3) 'matriz.h'
```
#ifndef   MATRIZ_H
#define   MATRIZ_H
 typedef float ** matriz;
 void crear(matriz * m , int n);
 float suma( matriz m, int n);
 void liberar ( matriz * m,  int n);
#endif
```

'matriz.c'
```
#include  <stdlib.h>
#include  "matriz.h"
void crear(matriz *m, int n){
    *m = (matriz) malloc (n* sizeof (float *));
    for (int i=0;  i<n;  i++){
        (*m)[i] = (float *) malloc (n* sizeof(float));
    }
}
float suma (matriz m,  int n){
    float suma =0;
    for (int i=0;  i<n;  i++){
        for(int j=0;  j<n;  j++){
            suma += m[i][j];
        }
    }
    return suma;
}
```

```c
void liberar(matriz *m , int n){
    for (int i=0; i<n ; i++){
        free((*m)[i]);
    }
    free(*m);
    *m = NULL;
}
```

:)

para que funque gcc -o main main.c matriz.c

4) a) Falso, &d.campo1
b) falso, ftell(archivo) devuelve pos actual de archivo
c) no dimos
d) falso, se posicionan al final
e) verdadero, si hacemos fprintf(stdout, "%s", string)

5) b) (lo único que corresponde al modulo 2) Falso, dado que para liberar memoria, antes hay que reservarla, y en todo caso sería free(p)

6) #define maxDos(a,b) ((a) > (b) ? (a) : (b))
#define maxCuatro(a,b,c,d) (maxDos(maxDos((a),(b)), maxDos((c),(d))))

7) Flag 0
Flag 0
Flag 0

HOJA N°

FECHA

'main.c'

8)
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main (int argc, char * argv[]){
    if (argc < 2){
        printf("error/n");
        return -1;
    }
    float promedio = 0.0;
    for(int i=1; i<argc; i++){
        promedio += atoi(argv[i]);
    }
    promedio/=(argc-1);
    printf("%.2f", promedio);
    return 0;
}
```

```
gcc -o main main.c
```

$./main\ \underbrace{1\ \ 2\ \ 3\ \ldots}_{\text{arguments}}$