UNIVERSIDAD DE MAGALLANES
FACULTAD DE INGENIERÍA
DEPTO. DE ING. EN COMPUTACIÓN

Punta Arenas, Julio 24 del 2003.

Programación de Computadores

Examen 2003

/ Nota: conteste cada pregunta en hojas separadas*/*

1.- *Cuestionario*

Para cada uno de los siguientes programas, imprima el resultado que aparece en pantalla,y haga un esquema de las declaraciones y asignaciones de las variables (dibujo como en clases). Si alguno de los codigos presenta error, indique donde y que es lo que ocurre. Además, asigne imaginariamente direcciones de memoria al comienzo del (o de los) programa(s) y muestre las direcciones finales de los punteros.

a)

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 5

int main()
{
   int i=0,a=15,b=35,*p,*q,*r, *m;
   q=(int *)malloc(MAX*sizeof(int));
   p=q;
   r=p;
   m=r
   for (i=0;i<MAX;i++,q++)
      *q=i*10;
   p=&a;
   q=&b;
   for (i=0;i<MAX;i++,r++)
      printf("%d, ",*r);
   printf("\n");
   p=m+2;

printf("%d,%d,%d,%d,%d,%d",a,b,*q,*r,*p,*m
);
printf("%p,%p,%p,%p,%p,%p",&a,&b,q,r,&p,&m
);

   return 0;
}
```

b)

```
#include <stdio.h>
#include <stdlib.h>

void quehace(int **bla, int **ble, int bli);
int main()
{
   int a=15,b=35, c=45,*p,*q;
   p=&a;
   q=&b;
   quehace(&p,&q,c);

   printf("%d,%d,%d,%d,%d",a,b,c,*p,*q);
   return 0;
}
void quehace(int **bla, int **ble, int bli)
{
   int *aux;
   aux=*bla;
   *bla=*ble;
   *ble=aux;
   bli++;
   printf("\ndentro de la funcion”);
   printf("%p,%p,%p",*bla,*ble,aux);
   printf("%p,%p,%p",bla,ble,aux);
   printf("\nSaliendo de la funcion”);
   return;
}
```

*(0.6 ptos.**a** y 0.4 ptos. **b**)*

2.- *Escriba una función para el cálculo de la suma de números imaginarios, para ello use la estructura siguiente:*

```
struct  _complex{
       float real;
       float imag;
       };
typedef struct _complex complex;
```

La función deben tener la siguiente cabecera, es decir, la función no retorna el resultado, si no lo guarda en la variable `resultado`.

```
void sumacomplex(complex *resultado, complex num1, complex num2);
```

(0.5 ptos)

<u>Supuesto</u> :

Dada las siguientes estructuras:

```
struct  _fecha{
       int anno;
       int mes;
       int dia;
       };
typedef struct _fecha fecha;
struct _ciudad{
       char codciudad[6];
       char nomciudad[20];
};
typedef struct _ciudad ciudad;
struct _persona{
       int codigo;
       char apellido[30];
       char nombre[25];
       int edad;
       fecha fnac;         /* fecha de nacimiento */
       char direccion[35];
       };
typedef struct _persona persona;
struct _vuelos{
       char codvuelo[6];
       char codsalida[6];  /*codigo ciudad*/
       char coddestino[6]; /*codigo ciudad*/
       long km;            /*Kilometros de vuelo entre ciudades*/
       };
typedef struct _vuelos vuelos;
struct _pasajes{           /* mantiene los viajes realizados por personas */
       int codigo;         /*codigo de la persona que vuela*/
       char codvuelo[6];
       fecha fecvuelo;
       };
typedef struct _pasajes pasajes;
```

Se tienen definidos los siguientes arreglos globales:

```
personas clientes[300];   /* arreglo que contiene los datos de cada
                             cliente */
vuelos [100];       /* arreglo que contiene los vuelos que realiza la
                             linea area.*/
```

```
pasajes viajes[10000];    /* arreglo que contiene el registro de los
                                pasajes vendidos (historico)*/
ciudad ciudades[30];      /*ciudades salidas y destinos*/
```

Considere que una persona puede volar varias veces entre dos ciudades iguales pero con distintas fechas de vuelo.

Problemas: cree las funciones necesarias para resolver las siguientes preguntas.

***3**.- Cual es el cliente que ha volado más veces entre Punta Arenas y Santiago (o viceversa), entre el 2000 (incluido) y hoy.*

(1.0 ptos.)

***4**.- Dado un arreglo auxiliar(local) del tipo:*

```
ciudad ciudadconsultas[10];
```

¿Cuantas peronas han volado a o desde esas ciudades?.

(0.5 ptos)

5.- *Mensajes Codificados*

```
#define MAX 11
struct  _cripto
{
   char data[MAX];  /*almacena hasta MAX-1 caracteres reales*/
   char parte1[MAX]; /*para otra cosa*/
};

typedef struct _cripto cripto;

typedef struct _jugador jugador;

int main()
{
   cripto frase;
   :
```

El código anterior es parte de la implementación simplificada de un sistema para codificar mensajes. El algoritmo es el siguiente:

- *Se van leyendo cada* `MAX-1` *caracteres del archivo original, el cual se ingresa a la estructura.*
- *Luego, se invierte el arreglo y se obtiene parte de la salida (puede ser a pantalla).*

Asuma que el archivo es direccionado como entrada (`a.out < mensaje`*).*
Ud. tiene que implementar el resto el programa.

(1.0 ptos.)

6*.- *Busca Minas

*Suponga que se le pide implementar <u>una parte</u> del juego llamado **Busca minas**, en el cual la idea es que se tiene un tablero de NxN celdas y donde se distribuyen M **bombas** (o minas), cada celda contígua a una bomba, debe indicar el número de bombas que tiene a sus lados (ver figura), si una celda no tiene bombas en alguno de sus lados, entonces su valor es 0.*

Ud. debe implementar la generación del tablero, es decir, debe ubicar M bombas en forma aleatoria en un tablero de NxN, puede usar N=11 y M=13.

Además, debe incluir los valores de las celdas donde no hay bombas, su tablero debería quedar como el de la figura (la cual es solo un ejemplo).

0	0	1	1	1	1	**B**	**B**	1	0	0
1	1	2	**B**	1	1	2	2	1	1	1
1	**B**	3	2	2	0	0	0	0	1	**B**
1	1	2	**B**	2	1	0	0	0	1	1
0	0	1	2	**B**	1	0	0	1	1	1
0	0	1	2	2	1	0	0	1	**B**	1
0	0	1	**B**	2	1	1	0	1	1	1
0	0	1	1	2	**B**	2	1	0	0	0
0	1	1	1	1	2	**B**	1	0	0	0
0	1	**B**	1	0	1	1	1	0	1	1
0	1	1	1	0	0	0	0	0	1	**B**

(2.0 ptos.)

Buena Caza

Profesor : ***Roberto Uribe P.***