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

Punta Arenas, 14 de Julio del 2003.

Programación de Computadores

Prueba Nº 3 (caps. 7, 8 y 9)

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

1.- ***Cuestionario***

Para cada uno de los siguientes programas, imprima el resultado que aparece en pantalla, preguntas a, b y d. Para la pregunta cescriba el código necesario para imprimir en pantalla la info del arreglo data, para ello sólo use el puntero 'p' (para recorrer e imprimir el arreglo) ..... NO USE data PARA IMPRIMIR!!.

a)

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

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

c)

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

#define MAX 5

struct test3
{
   int codigo;
   char nombre[20];
};

int main()
{
   struct test3 data[MAX], *p;
   int i=0;
   for (i=0;i<MAX;i++)
   {
      scanf("%d",&data[i].codigo);
      scanf("%s",data[i].nombre);
   }
/*
   escriba el codigo necesario para
...........

*/
   return 0;
}
```

d)

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 4

int main()
{
   int a=15,b=35,*p,*q,*r, *t;
   p=(int *)malloc(MAX*sizeof(int));
   q=&a;
   r=q;
   *q=10;
   t=r;
   *t=5;
   *r=0;
   *p=b;
   *(p+1)=*q;
   *(p+2)=*r;
   *(p+3)=*t;

   printf("%d,%d,%d,%d,%d,%d,%d",
*p,*(p+1),*(p+2),*(p+3),*q,*r,*t);
   return 0;
}
```

(0.2 ptos. a, b y d) (0.3 ptos c)

<u>Supuesto</u> :

Dada las siguientes estructuras:

```
struct  fecha{
       int anno;
       int mes;
       int dia;
       };
struct personal{
       int codigo;
       char apellido[30];
       char nombre[25];
       int edad;
       struct fecha fnac;        /* fecha de nacimiento */
       char direccion[35];
       int coddepto;
       };
struct depto{
       int coddepto;
       char nomdepto[35];
       };
struct sueldo{             /* mantiene los pagos de los empleados */
       int codigo;
       long sueldobase;
       long otros;
       long total;
       struct fecha fecpago;
       };
```

Se tienen definidos los siguientes arreglos globales:

```
struct personal obreros[100];    /* arreglo que contiene los datos de cada
                                 obrero */
struct depto ardepto[10];        /* arreglo que contiene el codigo y nombre de
                                 cada depto.*/
struct sueldo sueldo[10000];     /* arreglo que contiene los sueldos de los
                                 empleados, conserva los pagos historicos*/
```

Problemas:

2.- *Cree una función que, dado un nombre, apellido y un par de fechas, retorne la suma de los pagos entre esas fechas, los datos anteriores son pasados como parámetros a la función.*

(0.6 ptos.)

3.- *Escriba una función genérica que pueda responder a la siguiente pregunta. Quién fue la persona que gano más dinero entre los anhos 2000 y 2001 (incluidos)?. Es decir, dado un rango en años (los anteriores sólo son ejemplos), la función imprima en pantalla los datos de la persona, nombre, apellido y codigó.*

(1.0 ptos.)

4.- *Dado un año, imprima en pantalla un resumen del dinero gastado por departamento en sueldos.*

```
Código       Nombre Depto.            Total
2120         Finanzas                 $4500000.-
3124         Informatica              $6800000.-
:
:
```

(1.0 ptos.)

5.- ***Juego de Naipes***

Recuerde que para generar números aleatorios debe usar la función **`rand()`**, *el cual genera números entre 0 y* 2^{15} *-1. Además, tiene que inicializar con* **`srand(time(0))`** *al principio del programa para que los números no se repitan.*

Las funciones anteriores se encuentran en el archivo de cabecera **`stdlib.h`** *y* **`time.h`**..

Problema:

Suponga una ***baraja española****, la que tiene 40 cartas y 4 pintas (10 cartas de cada pinta).*

(Sota, caballo y Rey corresponden a los valores numéricos 8, 9 y 10 respectivamente)

El juego consiste en lo siguiente (algoritmo):

1. *4 jugadores con 3 cartas c/u.*
2. *Parte un jugador tirando una carta a la mesa (Ud. elige la primera vez).*
3. *Los demás jugadores deben tirar una carta de la misma pinta.*
4. *Gana esa mano el jugador que tira la carta más alta de esa pinta (gana 1 pto.).*
5. *Si no tiene de la pinta, tira cualquier otra carta (lo que implica perder esta vez).*
6. *Se repite el proceso desde el punto 2 (quedan dos cartas por jugador), pero ahora comienza el jugador que ganó la vez anterior.*
7. *Una vez que se tiraron las 3 cartas, se muestra el puntaje de cada jugador.*

Implemente el juego anterior. Para ello tome como base la siguiente estructura:

```
struct  carta
{
   int num;
   int pinta;
   char nompinta[10];  /*si fuese necesario*/
};
```

Implemente todas las estructura o arreglos necesarios para este juego.

<u>Ayuda:</u>

1.- Entre las diferentes estructuras, arreglos y variables, podría considerar una estructura que guardara la jugada actual, por ejemplo:

```
struct jugada
{
   int jugador;
   struct carta;
};
```

la que indicaría que jugador tiró que carta.

2.- Si quiere puede suponer que las cartas ya fueron almacenadas en alguna parte y las obtiene desde ahí, de todas maneras debe tener en cuenta los mecanismos para evitar que una carta aparezca dos veces.

(2.5 ptos.)

Buena Caza

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