Programación de Sistemas CCPG1008

Examen Parcial 2do Término 2018

Nombre:	Paralelo:

Pregunta 1 (25 puntos)

Considere una máquina de 64 bits, little endian. Muestre la salida del código mostrado abajo:

```
void see(int **dst, short *p, int *res, int ix, int i){
      *(*(dst+ix)) = *(p+i-ix-1);
      *res = *(*(dst+ix));
}
int main(void){
      int ta[10] = \{9,8,7,6,5,4,3,2,1,0\};
      short ti[6] = \{97, 42,33, 25,12, 29\};
      int *mi[6] = \{0\};
      printf("*****1*****\n");
      int *g = ta + 4;
      int i = 0;
      for(i = 0; i < 5; i++){
             *++g = *(ta+5+i)*2;
             printf("%d %d\n", i, *g);
      printf("*****2*****\n");
      int mem[6] = \{0\};
      int res = 0;
      for(i = 5; i>=0; i--){
             \star (mi+i) = &mem[5-i];
             see (mi, ti, &res, i, 6);
      }
      for(i = 0; i < 6; i++){
             printf("%d\n",*mi[i]);
      printf("*****3*****\n");
      unsigned char *p =(unsigned char*) &ta[2];
      printf("%02x\n", *p);
      printf("%02x\n", *(p+2));
      printf("%02x\n", *(p+4));
      return 0;
}
```

Pregunta 2 (20 puntos)

En el shell de Linux se ejecuta el siguiente comando para eliminar archivos de logs generados en el 2018:

```
rm -f log-???-2018-[0-1][0-9]-[0-3][0-9].*
```

Tache con una x los archivos que se eliminarían con este comando: (10 puntos)

¿Qué modificación debo hacer al comando para que, además del 2018, también borre los logs del 2015 (pero no 2016, 2017, 2014, 2013, etc.)? (10 puntos)

```
rm -f log-???-201[58]-[0-1][0-9]-[0-3][0-9].*
```

Pregunta 3 (18 puntos)

Asumiendo una arquitectura little endian de 64-bits y que los datos de una estructura son almacenados de manera contigua en memoria al igual que un arreglo, se tiene el siguiente código:

Si el valor de &dato es 0x100, muestre cómo se almacenaría en memoria (byte por byte) la variable dato marcando claramente la dirección de memoria de cada byte y su valor (en hexadcimal). Si el tipo de dato del byte es carácter, muestrelo como caracter entre comillas.

0 x 1 0	0 x 1 0	0 x 1 0 2	0 x 1 0 3	0 x 1 0 4	0 x 1 0 5	0 x 1 0 6	0 x 1 0 7	0 x 1 0 8	0 x 1 0 9	0 x 1 0 A	0 x 1 0 B	0 x 1 0 C	0 x 1 0 D	0 x 1 0 E	0 x 1 0 F	0 x 1 1	0 x 1 1	0 x 1 1	0 x 1 1 3	0 x 1 1 4	0 x 1 1 5	0 x 1 1 6	0 x 1 1 7	0 x 1 1 8
'S'	'B'	'P'	\0	'1'	'3'	'5'	'6'	'8'	'4'	'3'	'4'	'8'	'2'	\0	39	00	9C	01	00	00	00	00	00	00

Direcciones correctas: 3 puntos

Datos correctos: 10 Conversión correcta a hex datos y direcciones: 5 puntos

Pregunta 4 (25 puntos)

Muestre la salida del código mostrado abajo:

```
#define TAM 2
#define LENGTH 10
int main(void){
      for (int i = 0; i < 5; i++) {
            *(p+i) = i;
            \star (p+LENGTH-i-1) = \star (p+i) \star2;
            printf("%d\n", p[i]);
      }
      for(int i = 1; i \ge 0; i--){
            mx[i] = malloc(sizeof(short)*LENGTH);
      //mempcpy copia num bytes desde source a bloque de memoria destination
      //void * memcpy ( void * destination, const void * source, size t num );
            memcpy(mx[i], p, LENGTH*sizeof(short));
            for (int j = 0; j < LENGTH; j++)
                  mx[i][j] = mx[i][j] * i;
      for(int i = 0; i < TAM; i++){</pre>
            for (int j = 0; j < LENGTH; j++)
                  printf("%2d ",mx[i][j]);
            printf("\n");
      printf("\n");
      //realloc cambia el tamaño del bloque de memoria asignado por malloc.
      //void *realloc( void *ptr, size_t new_size );
      mx = realloc(mx,sizeof(short *)*2*TAM); //****3******
      for (int i = 2*TAM-1; i \ge 0; i--) {
            if(i > 1){
                  mx[i] = malloc(sizeof(short)*LENGTH);
                  memcpy(mx[i], p, LENGTH*sizeof(short));
                  for (int j = 0; j < LENGTH; j++)
                        mx[i][j] = mx[i][j] * i;
            }
      for (int i = 0; i < 2*TAM; i++) {
            for (int j = 0; j < LENGTH; j++)
                  printf("%2d ",mx[i][j]);
            printf("\n");
      return 0;
}
```

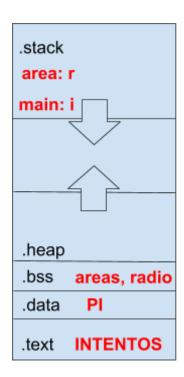
```
0  (Parte 1: 7 puntos)
1
2
3
4
          (Parte 2: 8 puntos)
0  0  0  0  0  0  0  0  0
0  1  2  3  4  8  6  4  2  0

          (Parte 3: 10 puntos)
0  0  0  0  0  0  0  0  0
0  1  2  3  4  8  6  4  2  0
0  2  4  6  8  16  12  8  4  0
0  3  6  9  12  24  18  12  6  0
```

Pregunta 5 (12 puntos)

```
#define INTENTOS 3
float areas[INTENTOS];
float PI = 3.14159;
float area(float r)
{
      return PI*r*r;
}
int main(){
      static float radio;
      for(int i = 1; i <= INTENTOS; i++)</pre>
            printf("Intento %d) Ingrese el radio: ", i);
            scanf("%f",&radio);
            areas[i-1] = area(radio);
            printf("El area del círculo es %f\n", areas[i-1]);
      }
}
```

Posicione los símbolos del código arriba mostrado: **INTENTOS**, **areas**, **PI**, **r**, **radio**, **i** en el siguiente diagrama de la memoria virtual del proceso. Si posiciona símbolos en el stack, indique de qué función.



2 puntos por símbolo