Examen de Mejoramiento de Programación de Sistemas

1er término 2017 – 2018

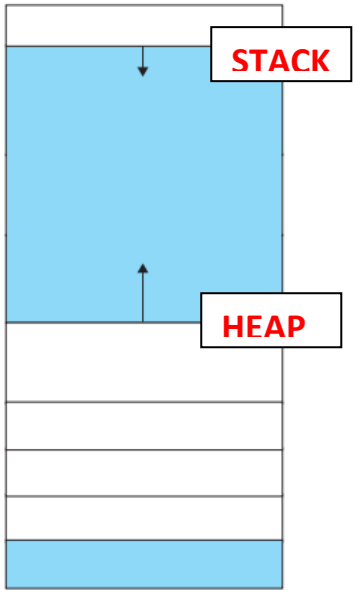
Nombre: _____

Pregunta 1 (10 puntos) PREGUNTA ANULADA

Considere que el siguiente programa corre en una máquina Big Endian. ¿Cuál es la salida del programa?

```c
#include <stdio.h>

typedef struct tda{
	int w;
	char *m;
}est;

void fn(est *p){

	p->w = 11;
	p->m = "final";

	unsigned short *pt = (unsigned short *)&p->w;

	printf("%hu\n",*pt);
	printf("%hu\n",*(pt+1));
	printf("%hu\n",*(pt+2));
	printf("%hu\n",*(pt+3));
}

int main(void){
	est s;

	s.w = 9;
	s.m = "mejoramiento";

	fn(&s);

	printf("%d\n",s.w);
	printf("%s\n",s.m);
}
```

Pregunta 2 (10 puntos)



a) Escriba en el diagrama anterior la ubicación del **heap** y del **stack**
b) Dada la siguiente función, ¿qué variables se almacenan en el **heap** y cuáles en el **stack**?

El contenido al que apunta **t** se almacena en el HEAP, sin embargo el puntero **t** se almacena en el STACK.

w, s y **j** se almacenan en el STACK.

No se puede saber si el contenido al que apunta **m** está en el STACK o en el HEAP.

```
int fn(char w, int s, char *m){
      int j = s;
      char *t = (char *)malloc(100);

      ...

      return j;
}
```

Pregunta 3 (5 puntos)

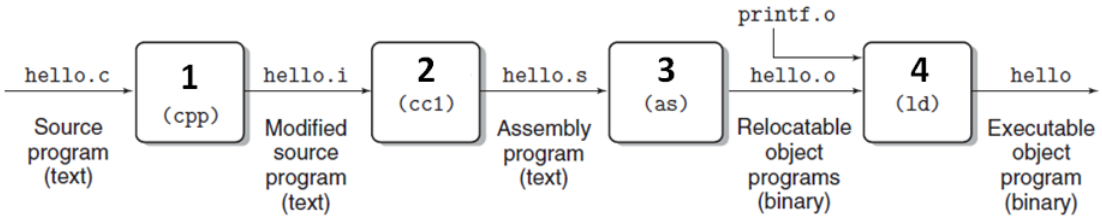
Considere un número **entero de 8 bits con signo**. ¿Cuál es el número más pequeño y más grande que puede representar este tipo de dato?

Más pequeño (o menor): -128

Más grande (o mayor): 127

Pregunta 4 (15 puntos)

Identifique y explique (en una línea o dos) cada una de las etapas del sistema de compilación.



1. Preprocesador: Modifica el código original y reemplaza las directivas que empiezan con #.
2. Compilación: Convierte el código en C entrante a código ensamblador en forma de texto.
3. Ensamblador: Convierte el código ensamblador a código de máquina. El resultado es un archivo binario con códigos de instrucciones específicos para la arquitectura destino.
4. Linking: Enlaza las librerías estáticas y dinámicas al código de máquina entrante y crea un archivo listo para ser ejecutado en la arquitectura destino.

Pregunta 5 (10 puntos)

Explique en detalle el funcionamiento de la siguiente llamada a open:

```
fd = open("foo.txt", O_RDWR|O_CREAT|O_APPEND, 0);
```

En esta llamada se obtiene un descriptor de archivo al archivo foo.txt. Con este descriptor es posible leer y escribir en el archivo (O_RDWR), si el archivo no existe es creado automáticamente (O_CREAT) y los datos que se escriben en el archivo se agregan al final del mismo, es decir, no se sobrescribe la información ya existente en el mismo (O_APPEND).

Pregunta 6 (15 puntos)

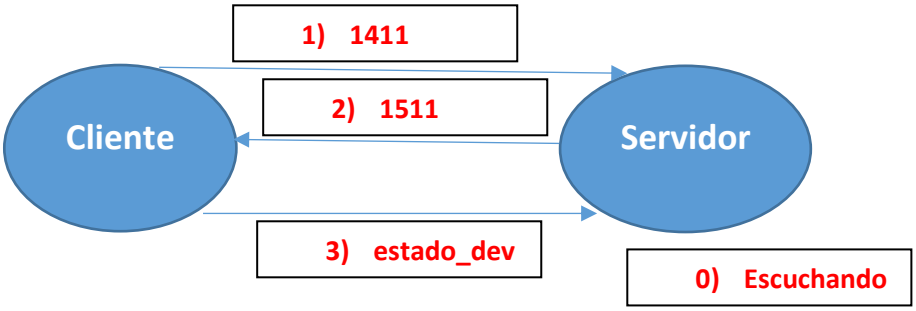
Determine cuántos “hello” muestra este programa al ejecutarse.

```
1   #include "csapp.h"
2
3   void doit()
4   {
5       Fork();
6       Fork();
7       printf("hello\n");
8       return;
9   }
10
11  int main()
12  {
13      doit();
14      printf("hello\n");
15      exit(0);
16  }
```

Debido a que hay dos fork(), existen 4 procesos. Cada proceso imprime *hello* dos veces por lo tanto el programa muestra 8 veces “hello” al ejecutarse.

Pregunta 7 (20 puntos)

Dada la siguiente implementación de comunicación entre cliente y servidor usando sockets. Dibuje un diagrama del protocolo utilizado para el envío del valor de "*estado_dev*", indicando el orden y sentido en el que se lleva la comunicación, justificando posteriormente su respuesta. (Asuma equivalencia entre: *read* y *recv*; *write* y *send*)



0) El servidor escucha nuevas conexiones.

1) El cliente se conecta y envía el código: *1411*

2) El servidor acepta la conexión, verifica la recepción de *1411* y envía el código *1511*.

3) El cliente verifica la recepción de *1511* y envía el valor de la variable *estado_dev*.

```c
/***Servidor ***/
#define HS_MSG 30
#define MAXDATA 255

char HS_msg[HS_MSG];

listenn = listen(MySocket, BACKLOG);

while(1)
{
  tamano = sizeof(struct sockaddr_in);

  /***/
  NewSocket = accept(
        MySocket,
        (struct sockaddr *)&direction,
        (socklen_t *)&tamano);
  /***/

  memset(HS_msg, 0, HS_MSG);

  recv(NewSocket,
    HS_msg,
    HS_MSG, 0);

  if(strcmp(HS_msg,"1411")==0)
  {
    memset(HS_msg, 0, HS_MSG);
    sprintf(HS_msg,"1511");

    send(NewSocket,
      HS_msg,
      HS_MSG, 0);
  }
  else
    continue;

  /***/
  if (!fork())
  {
    memset(Mensaje.buffer_msg, 0, MAXDATA);
    recv(
      NewSocket,
      Mensaje.buffer_msg,
      MAXDATA, 0);

    close(NewSocket);
    num_conexiones--;
  }
  /***/
  close(NewSocket);

  while(waitpid(-1,NULL,WNOHANG) > 0);
}
```

```c
/*** Cliente ***/
int envia_msg(char *serverIP, int
estado_dev)
{
  char buf[MAXDATA];
  char msg[HS_MSG];
  hostent *hostt;

  sprintf(ip,"%s",serverIP);
  hostt = gethostbyname(ip);

  MySocket = socket(AF_INET,
        SOCK_STREAM, 0);

  su_direccion.sin_family = AF_INET;
  su_direccion.sin_port = htons(PORTA);
  su_direccion.sin_addr =
    *((struct in_addr *)hostt->h_addr);
  bzero(&(su_direccion.sin_zero), 8);

  connect(
    MySocket,
    (struct sockaddr *)&su_direccion,
    sizeof(struct sockaddr));

  memset(msg, 0, HS_MSG);
  sprintf(msg,"1411");

  send(MySocket,
    msg,
    HS_MSG, 0);

  memset(msg, 0, HS_MSG);
  recv(MySocket,
    msg,
    HS_MSG, 0);

  if(strcmp(msg,"1511")==0)
  {
    memset(buf, 0, MAXDATA);
    sprintf(buf,"%d",estado_dev);
    send(MySocket,
      buf,
      HS_MSG, 0);
  }

  close(MySocket );

  return 0;
}
```

Pregunta 8 (15 puntos)

Determine la salida del siguiente código:

```
struct arg_struct {
  int arg1;
  int arg2;
};

void *Hilo1(void *argus)
{
  struct arg_struct *args = (struct arg_struct *) argus;
  int cont=15;

  while (cont > 0)
    cont=cont-args->arg1;

  printf("Cont Hilo1: %d", cont);
  pthread_exit(NULL);
  return NULL;
}

void *Hilo2(void *argus)
{
  struct arg_struct *args = (struct arg_struct *) argus;
  int cont=15;

  while (cont > 0)
    cont=cont-args->arg2;

  printf("Cont Hilo2: %d", cont);
  pthread_exit(NULL);
  return NULL;
}


int main()
{
  pthread_t myThread1, myThread2;
  struct arg_struct args;

  args.arg1 = 5;
  args.arg2 = 7;

  pthread_create(&myThread1, NULL, &Hilo1, (void *)&args);
  pthread_create(&myThread2, NULL, &Hilo2, (void *)&args);

  pthread_join(myThread1, NULL);
  pthread_join(myThread2, NULL);

  return 0;
}
```

Cont Hilo2: -6 Cont Hilo1: 0