Trabajo Colaborativo - Sesión 9(3)
Prueba de Software

1. Propósito:

1.1. Realizar pruebas de caja negra o blanca.

2. Indicaciones/instrucciones:

2.1. Construya el grafo de flujo para el pseudocódigo adjunto, calcule la complejidad ciclomática, el conjunto de caminos básico, y casos de prueba de caja negra asociados.

```
INICIO
  Leer_Nota_de_Teoría (NT);
  Leer_Nota_de_Prácticas (NP);
  Leer_Nota_de_Trabajos (NTR);
  SI NP = No Apto
     ENTONCES NF = 4;
  SINO
     SI NT<4.5
        ENTONCES NF=NT
     SINO NF = NT + NTR;
     FINSI
     SI NF>10
        ENTONCES NF = Matricula de Honor;
     FINSI
  FINSI
FIN
```

2.2. Dado el siguiente pseudocódigo construya: el grado de flujo, la complejidad ciclomática, el conjunto de caminos básico y casos de prueba de caja negra asociados.

```
INICIO;
     LEER (a);  LEER (b);
     t = 0;
     SI  (a < b) ENTONES
         MIENTRAS (a > 0) HACER
               t = t + b;  a = a  - 1;
         FinMIENTRAS
     SI NO
         MIENTRAS (b > 0) HACER
               t = t + a;  b = b  - 1;
         FinMIENTRAS;
     FinSI
     ESCRIBIR (t);
FIN;
```

2.3. A partir del código siguiente y utilizando un grafo de flujo, construya la complejidad ciclomática y el conjunto de caminos básico; y casos de prueba de caja negra asociados.

```
#define nombrefichero "numeros"
/* Calcula la media de los numeros positivos contenidos en un archivo*/
void CalcularMedia( float *media )
{ FILE *fichero;
  int  valor, numvalores, suma;
 *media = 0.0;
  numvalores = 0; suma = 0;
  if ((fichero = fopen(nombrefichero, "rb"))==NULL)
   { printf("\nEl fichero no se encuentra\n");
         return; }
  while (fread(&valor, sizeof(valor), 1, fichero))
       {  if (valor>0)
                 { numvalores = numvalores + 1;
                suma = suma + valor;
              }
       }
  /* calcular la media e imprimir resultados */
  if (numvalores>0)
      { *media = suma/numvalores;
     printf("\n Se leyeron %d: \n", numvalores);
   }
  else printf("\nNo se encontraron valores positivos\n");
  fclose(fichero);
}
```

2.4. Dado el siguiente código en C, especifique un conjunto de casos de prueba mediante la técnica de caja negra. Defina el grafo de flujo y la complejidad ciclomática.

Cod_Película	Título	Formato	Alquilada
00001	"Star Wars"	DVD	SÍ
00002	"Star Wars"	VHS	SÍ
00003	"Matrix"	DVD	SÍ
00004	"Matrix"	VHS	NO

```
int Obtener_Pelicula_Disponible (char cod_pelicula[5])
{
  char formato[10], resp;
  int disponible=0, formato_valido=0;
  do
  {
   LEER_FORMATO(formato);
      formato_valido= VALIDAR_FORMATO(formato);
   if (!formato_valido)
       return (FORMATO_NO_EXISTENTE);
      else
      {
      LEER_TITULO_PELICULA(titulo);
      if (! EXISTE_PELICULA(titulo))
          return (PELICULA_NO_EXISTENTE);
      else
       {
        disponible= VALIDAR_PELICULA_DISPONIBLE (formato, titulo, cod_pelicula);
        if (!disponible)
           return (PELICULA_NO_DISPONIBLE);
        else
         {
          printf("Ha  escogido  la  pelicula\n  %s,  en  formato  %s",  titulo,
formato);
          printf("¿Está seguro? (s/n)\n");
          scanf("%c",&resp);
         }
       }
      }
   } while (resp!='s');
 }
```

2.5. Dado la siguiente función en C, construir el grafo de flujo, determinar un conjunto básico de caminos y un conjunto de casos de prueba (en caja negra) para dicho conjunto básico.

```
int buscar_en(char cadena[10], char letra)
// la función busca si una letra aparece en una determinada cadena
// entradas:  cadena : la cadena donde se realiza la búsqueda
//            letra:  el carácter a buscar
// devuelve el número de ocurrencias de la letra.

{ int contador, n,  lon;
  n=0; contador=0;
  lon = strlen (cadena);  // devuelve la longitud de la cadena
  if (lon > 0) {
      do {
       if (cadena[contador] == letra) {
                  n++;      // incrementa el número de ocurrencias
                }
       contador++;     // pasa a la siguiente letra de la cadena
       lon--; // disminuye el tamaño de la cadena que queda por revisar
      } while (lon > 0);
   }
      return n;
}
```

2.6. Dada la siguiente función en C, construye el grafo de flujo, deriva un conjunto de caminos básicos y define un conjunto de casos de prueba en caja negra para dicho conjunto básico.

```
int hay_mayor_tira (char tira [4], char letra)
//la función comprueba si en tira hay un carácter mayor que letra
// entradas: tira : la cadena donde se realiza la comprobación
//           letra : carácter a comparar

{
  int encontrado,j,lon;
  encontrado=0; j=0;
  lon=strlen(tira);  // devuelve la longitud de la cadena
  while ((!encontrado) && (j<lon))
    {
      if (tira[j]>letra) encontrado=1;      // ha encontrado un carácter mayor
      j++;           // pasa a la siguiente letra de la cadena
    }
  return encontrado;
}
```

2.7. Construya los casos de prueba, utilizando la técnica del camino básico para la siguiente función.

```
int validar_cliente_existe (Codigo_Cliente cc)
/* Entrada: cc, Código de Cliente a buscar
   Salida: 1 si cc existe; 0 si no existe;
          -1 si no encuentra el archivo
*/
{
   FILE *ArchivoClientes;
   int encontrado = 0;
  int fin_archivo = 0;
   Reg_Cliente rc;
   if ((ArchivoClientes=fopen("Clientes","rb"))==NULL) return -1;
   else {
     fin_archivo = fread((&rc,sizeof(rc),1,ArchivoClientes);
     while (!fin_archivo){
       if (!strcmp(rc.cod_cliente,cc))
         return 1;
       fin_archivo = fread(&rc,sizeof(rc),1,ArchivoClientes);
     }
     fclose(ArchivoClientes);
     return 0;
   }
}
```

2.8. Construya los casos de prueba de caja negra para la siguiente función.

```
int sum_vect(int v[])
{
int i=0;
int suma=0;
while (i<2)
{
     if (v[i]==0) then break;
     suma = suma + v[i];
     i = i+1;
}
return suma;
}
```

2.9. Construya el grafo de flujo para la siguiente función, calcule la complejidad ciclomática, el conjunto de caminos básico y los casos de prueba en caja negra.

```
bool es_primo(int n) {
   if (n <= 0 || n == 1) return false;
   for (int i = 2; i <= n/2; ++i)
      if (n % i == 0) return false;
   return true;
}
```