PRÁCTICA 1. PERIFÉRICOS Y DISPOSITIVOS INTERFAZ HUMANA

Antonio Jesús Ruiz Gómez.

1. Diseñar e implementar 9 funciones básicas de Entrada/Salida a través de llamadas a interrupciones.

Función 1 → gotoxy() → coloca el cursor en una posición determinada

```
//Funcion gotoxy()
//coloca el cursor en una posición determinada
void gotoxy(int x, int y){
  union REGS inregs, outregs;

  inregs.h.dh=y;
  inregs.h.dl=x;

  inregs.h.ah=0x02;
  inregs.h.bh=0x00;

  int86(0x10, &inregs, &outregs);
}
```

Función 2 → setcursortype() → fijar el aspecto del cursor, debe admitir tres valores: INVISIBLE, NORMAL y GRUESO.

```
void setcursortype(int tipo){

  union REGS inregs, outregs;

  switch (tipo)
  {

    case 1:
        inregs.h.ch=010;
        inregs.h.cl=000;
        break;
    case 2:
        inregs.h.ch=010;
        inregs.h.cl=010;
        break;
    case 3:
        inregs.h.ch=000;
        inregs.h.cl=010;
        break;
    default:
        printf("Tipo no permitido ");

  }

  inregs.h.ah=0x01;

  int86(0x10, &inregs, &outregs);

}
```

Función 3 → setvidemode() → fija el modo de video deseado

```
//Funcion setvidemode()
//fija el modo de video deseado
void setvideomode(BYTE mode){
  union REGS inregs, outregs;

  inregs.h.ah = 0x00;
  inregs.h.al = mode;

  int86(0x10,&inregs,&outregs);
}
```

Función 4 → getvidemode() → obtiene el modo de video actual

```
//Funcion getvideomode()
//obtiene el modo de video actual
int getvideomode(){
  union REGS inregs, outregs;

  inregs.h.ah = 0x0F;

  int86(0x10,&inregs,&outregs);

  return outregs.h.al;
}
```

Función 5 → textcolor() → modifica el color de primer plano con que se mostrarán los caracteres

```
void textcolor(int color){
  letra=color;
}
```

Función 6 → textbackground() → modifica el color de fondo con que se mostrarán los caracteres

```
void textbackground(int color){
  fondo=color;
}
```

Función 7 → clrscr() → borra toda la pantalla

```
void clrsrc(){
  union REGS inregs, outregs;

  inregs.h.ah = 0x06;
  inregs.h.al = 0x00;
  inregs.h.bh = fondo_c;
  inregs.h.ch = 0x00;
  inregs.h.cl = 0x00;
  inregs.h.dh = getfilas()-1;
  inregs.h.dl = getcolumnas()-1;

  int86(0x10,&inregs,&outregs);

  gotoxy(0,getfilas());

}
```

Función 8 → cputchar() → escribe un carácter en pantalla con el color indicado actualmente

```
void cputchar(char caracter){
  union REGS inregs, outregs;

  BYTE color_fondo_byte;
  BYTE color_final;

  color_fondo_byte = fondo;

  color_fondo_byte = color_fondo_byte << 4;

  color_final = color_fondo_byte | letra;

  inregs.h.ah = 0x09;
  inregs.h.al = caracter;
  inregs.h.bl = color_final;
  inregs.h.bh = 0x00;
  inregs.x.cx = 1;

  int86(0x10, &inregs, &outregs);
}
```

Función 9 → getche() → obtiene un carácter de teclado y lo muestra en pantalla

```
void getche(int imprimir){
  union REGS inregs, outregs;
  char caracter;

  inregs.h.ah = 0x01;

  int86(0x21, &inregs, &outregs);

  caracter = outregs.h.al;

  if(imprimir == 1){
    printf("\nCaracter pulsado: ");
    cputchar(caracter);
  }
}
```