

Práctica 1 PDIH

Ángel López Martos

Requisitos mínimos:

-mi_getchar()

```
int mi_getchar(){  
  
    union REGS inregs, outregs;  
    int character;  
    inregs.h.ah = 1;  
    int86(0x21, &inregs, &outregs);  
    character = outregs.h.al;  
    return character;  
}
```

-mi_putchar(char c)

```
void mi_putchar(char c){  
  
    union REGS inregs, outregs;  
    inregs.h.ah = 2;  
    inregs.h.dl = c;  
    int86(0x21, &inregs, &outregs);  
}
```

-mi_gotoxy(int x, int y)

```
void mi_gotoxy(int x, int y){ /*Coloca el cursor en una posición determinada*/  
  
    union REGS inregs, outregs;  
  
    inregs.h.ah = 2;        //Numero de función  
    inregs.h.dh = x;        //Numero de fila  
    inregs.h.dl = y; //Numero de columna  
    inregs.h.bh = 0;        //Preguntar??  
  
    int86(0x10, &inregs, &outregs);  
  
}
```

-setcursortype(int tipo_cursor)

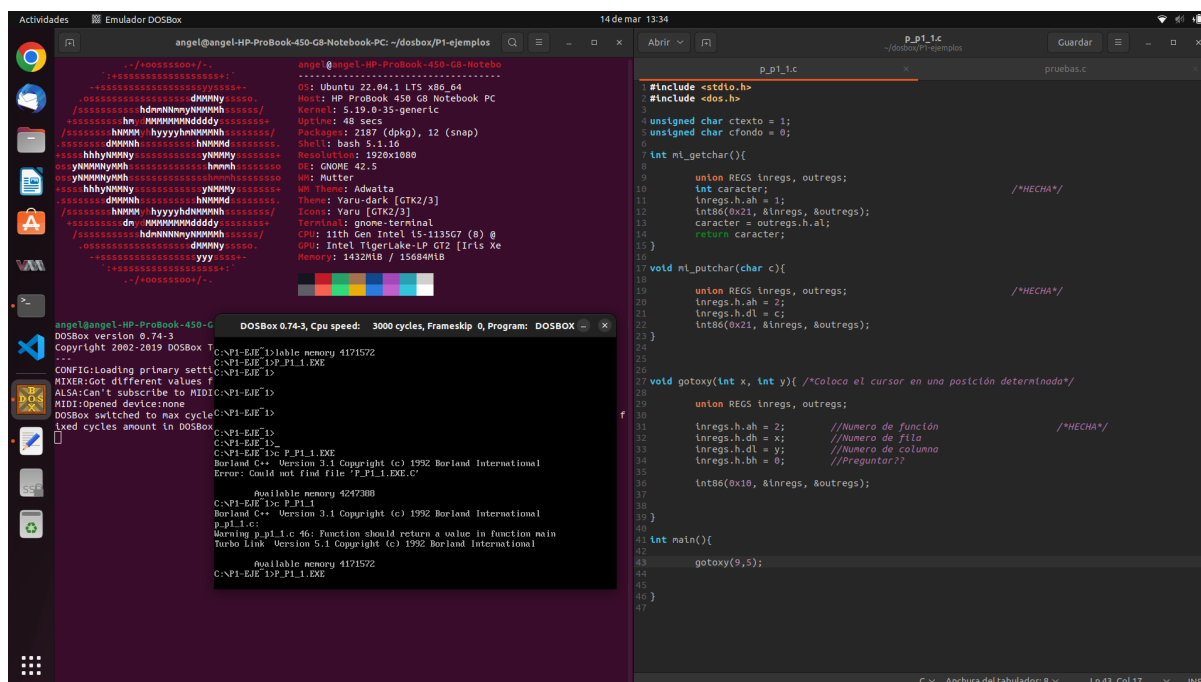
void setcursortype(int tipo_cursor){/*fijar el aspecto del cursor, debe admitir tres valores INVISIBLE, NORMAL y GRUESO*/

```
    union REGS inregs, outregs;
    inregs.h.ah = 0x01;

    switch(tipo_cursor){
        case 0: //invisible
            inregs.h.ch = 010;
            inregs.h.cl = 000;
            break;
        case 1: //normal
            inregs.h.ch = 010;
            inregs.h.cl = 010;
            break;
        case 2: //grueso
            inregs.h.ch = 000;
            inregs.h.cl = 010;
            break;
    }
```

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

```
}
```



-setvideomode(unsigned char modo)

```
void setvideomode(unsigned char modo){ /*fijar el modo de video deseado, 3-texto, 4 gráfico*/

    union REGS inregs, outregs;
    inregs.h.al = modo;
    inregs.h.ah = 0x00;
    int86(0x10, &inregs, &outregs);

}
```

-getvideomode(void)

```
unsigned int getvideomode(void){ /*obtiene el modo de video actual*/

    union REGS inregs, outregs;
    inregs.h.ah = 0x0F; //Función Fh
    int86(0x10, &inregs, &outregs); //Numero de interrupción
    //int res = outregs.h.al; //Por algún motivo no me deja copiar el valor en una variable
    //return res;

    return outregs.h.al;

}
```

-textcolor()

```
void textcolor(){/*modifica el color del primer plano con el que se mostrarán los caracteres*/

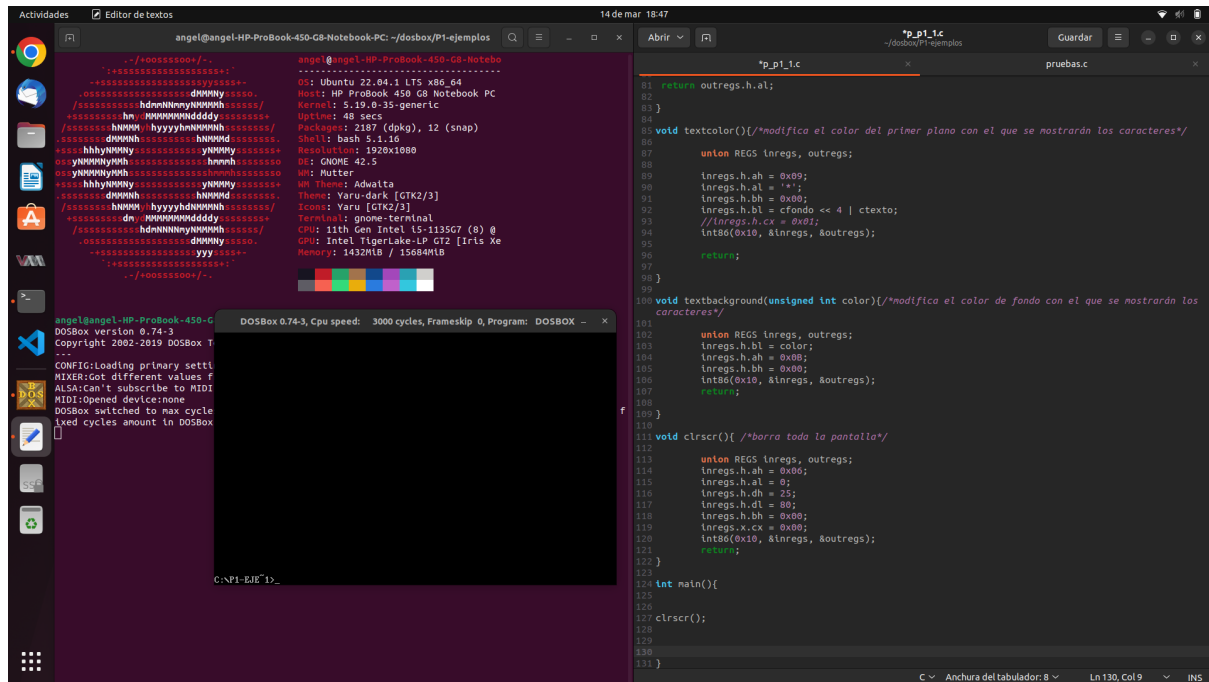
    union REGS inregs, outregs;

    inregs.h.ah = 0x09; //Por algún motivo no funciona para mas de un char*/
    inregs.h.al = '*';
    inregs.h.bh = 0x00;
    inregs.h.bl = cfondo << 4 | ctexto;
    inregs.x.cx = 0x01;
    int86(0x10, &inregs, &outregs);

    return;

}
```

-textbackground(unsigned int color)



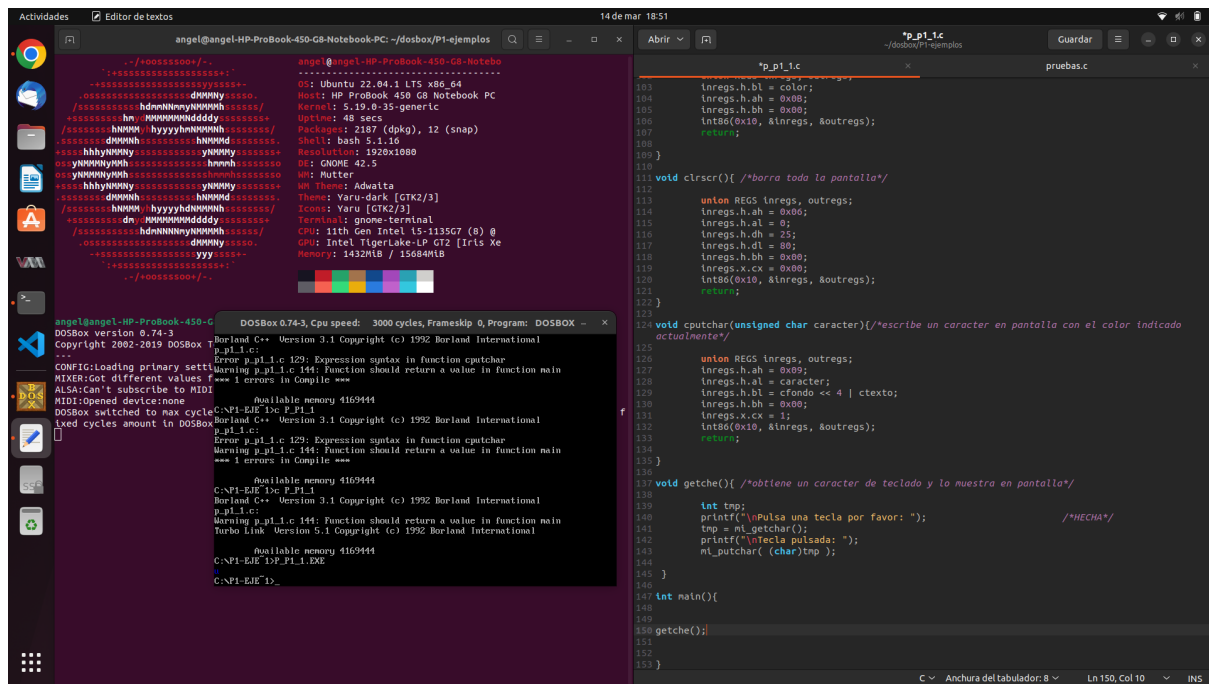
-mi_cputchat(unsigned char character)

void mi_cputchat(unsigned char character){/*escribe un caracter en pantalla con el color indicado actualmente*/

```

    union REGS inregs, outregs;
    inregs.h.ah = 0x09;
    inregs.h.al = character;
    inregs.h.bl = cfondo << 4 | ctexto;
    inregs.h.bh = 0x00;
    inregs.x.cx = 1;
    int86(0x10, &inregs, &outregs);
    return;
}

```

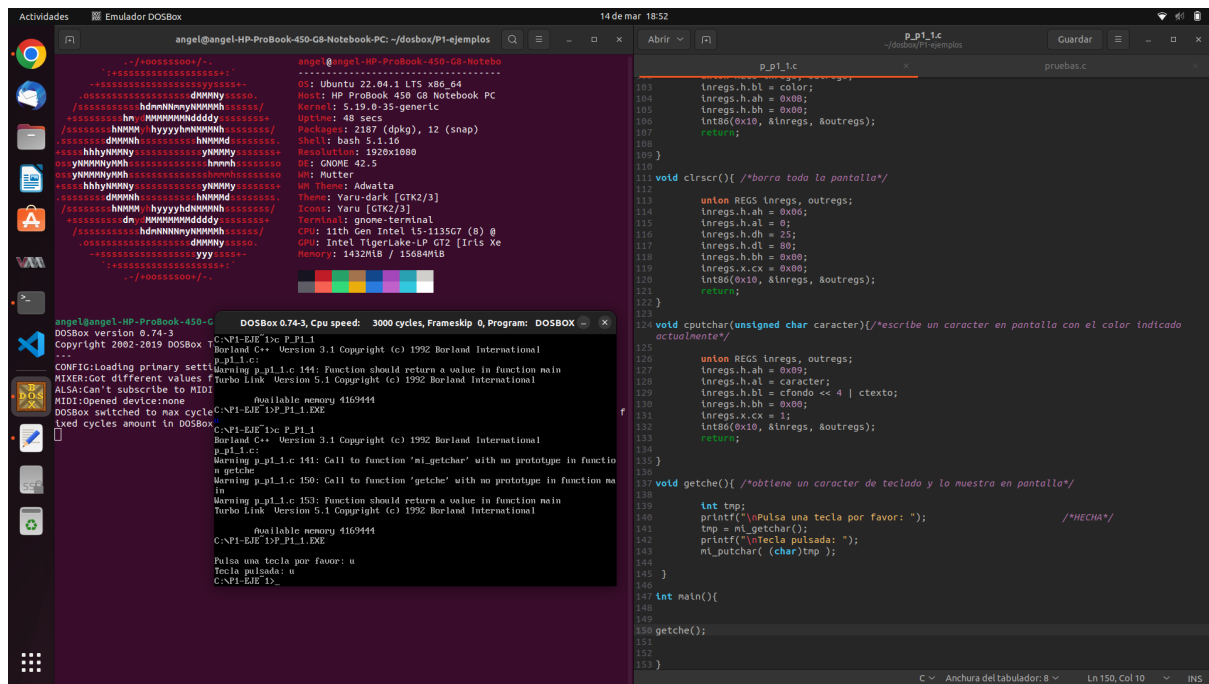


-getche())

```
void getche(){ /*obtiene un caracter de teclado y lo muestra en pantalla*/
```

```
    int tmp;
    printf("\nPulsa una tecla por favor: ");          /*HECHA*/
    tmp = mi_getchar();
    printf("\nTecla pulsada: ");
    mi_putchar( (char)tmp );
```

```
}
```



-mi_textcolor(int color)

```
void mi_textcolor(int color){
    ctexto = color;
}
```

-mi_textbackground(int color)

```
void mi_textbackground(int color){
    cfondo = color;
}
```

-dibuja_cuadrado(int superior_izq_x, int superior_izq_y, int inferior_dcha_x, int inferior_dcha_y, unsigned char ctexto, unsigned char cfondo)

```
void dibuja_cuadrado(int superior_izq_x, int superior_izq_y, int inferior_dcha_x, int inferior_dcha_y, unsigned char ctexto, unsigned char cfondo){
```

```
    int i,j,k;
    char espacio = ' ';
```

```

//Cambia de color para dibujar
mi_textcolor(ctexto);
mi_textbackground(cfondo);

//tapa
for(i = superior_izq_x; i<inferior_dcha_x-1; i++){
    mi_gotoxy(i,superior_izq_y);
    mi_cputchar(espacio);
}

//lados
for(k = superior_izq_y; k<inferior_dcha_y; k++){
    mi_gotoxy(inferior_dcha_x-1,k);
    mi_cputchar(espacio);
    mi_gotoxy(superior_izq_x,k);
    mi_cputchar(espacio);
}

//fondo
for(j = superior_izq_x; j<inferior_dcha_x-1; j++){
    mi_gotoxy(j,inferior_dcha_y-1);
    mi_cputchar(espacio);
}
}

```

The screenshot shows a DOSBox emulator window titled "Emulador DOSBox" with a file explorer view of "Carpeta personal / dosbox / P1-ejemplos". The terminal window displays a colorful ASCII art of a face with the following text:

```

WM Theme: Adwaita
Theme: Varu-dark [GTK2/3]
Icons: Varu [GTK2/3]
Terminal: gnome-terminal
CPU: 11th Gen Intel i5-1135G7 (8) @
GPU: Intel TigerLake-LP GT2 [Iris Xe
Memor.: 2517MiB / 15684MiB

```

Below the terminal, a small window shows a red rectangle. To the right, a code editor window titled "p_p1-1.c" shows the following C++ code:

```

172 //Dibujar el cuadrado
173
174 }
175
176 //Modifica el color de primer plano con que se mostrarán los caracteres
177 } */
178
179 // modifica el color de primer plano con que se mostrarán los caracteres
180 void mi_textcolor(int color){
181     ctexto = color;
182 }
183
184 void mi_textbackground(int color){
185     cfondo = color;
186 }
187
188 void dibuja_cuadrado(int l_x, int l_y, int r_x, int r_y, unsigned char ctexto, unsigned char
cfondo){
189
190     int i;
191     char myChar = ' ';
192
193     //Cambiar color del texto
194     mi_textcolor(ctexto);
195     mi_textbackground(cfondo);
196
197     //Dibujar arista superior
198     for(i = l_x; i<r_x-1; i++){
199         mi_gotoxy(i,l_y);
200         mi_cputchar(myChar);
201     }
202
203     //Dibujar laterales
204     for(i = l_y; i<r_y-1; i++){
205         mi_gotoxy(l_x,i);
206         mi_cputchar(myChar);
207         mi_gotoxy(r_x,i);
208         mi_cputchar(myChar);
209     }
210
211     //Dibujar arista inferior
212     for(i = l_x; i<r_x-1; i++){
213         mi_gotoxy(i,r_y-1);
214         mi_cputchar(myChar);
215     }
216 }
217
218 int main(){
219
220     clrscr();
221     dibuja_cuadrado(5,10,10,2,4);
222
223
224 }

```