Saptamana 8: Procese

Să se modifice programul din săptămâna anterioară astfel încât acesta va primi un al doilea argument ce va reprezenta calea către un director și prin urmare se va apela astfel:

```
./program <director_intrare> <director_iesire>
```

Pentru fiecare intrare (fișier obișnuit, fișier `.bmp`, director, legătură simbolică ce indică spre un fișier obișnuit) din directorul de intrare (primul argument), procesul părinte va crea câte un nou proces care va scrie informațiile referitoare la statistică într-un fișier din directorul de ieșire (al doilea argument), cu numele `<nume_intrare>_statistica.txt`, unde `<nume_intrare>` reprezinta numele intrarii curente din directorul de intrare.

La încheierea fiecărui proces fiu, acesta va trimite părintelui numărul de linii scrise în fișierul `statistica.txt`.

În plus, pentru fiecare intrare ce reprezinta o imagine `.bmp` procesul părinte va crea un al doilea proces (pe langa cel responsabil de scrierea de informații în fișierul de statistică), care sa citească întreg conținutul fișierului și care va converti imaginea în tonuri de gri. Știind ca în cadrul imaginilor `.bmp` fiecare pixel este caracterizat de 3 valori cuprinse între 0 și 255 (reprezentate pe 1 octet), puteți folosind formula de mai jos pentru conversia cerută:

P_gri = 0.299 * P_rosu + 0.587 * P_verde + 0.114 * P_albastru

unde P_rosu, P_verde, respectiv P_albastru, reprezinta intensitatile celor trei culori.

La încheierea fiecărui proces fiu, părintele îi va prelua starea și va afișa un mesaj de forma "`S-a încheiat procesul cu pid-ul <PID> și codul <cod>`", unde `<PID>` reprezinta PID-ul procesului fiu care s-a incheiat, iar `<cod>` reprezinta codul cu care acesta s-a incheiat.

!!! TOATE PROCESELE CREATE TREBUIE SA RULEZE IN PARALEL !!!

Informații adiționale referitoare la prelucrarea pixelilor:

- Structura unui fișier `.bmp` este prezentată mai jos.
- Valorile pe care trebuie sa le modificati se găsesc în zona denumite Raster Data.
- Numărul total de pixeli este **height x width**, unde height reprezinta inaltimea imaginii, iar width reprezinta latimea imaginii (ambele valori putand fi obținute din info header).
- Tineti cont ca fiecare pixel este reprezentat cu ajutorul a 3 valori, fiecare reprezentată pe cate un octet.

- Pentru a obține efectul cerut va trebui sa suprascrieti fiecare din cele 3 valori corespunzătoare unui pixel, P_rosu, P_verde, respectiv P_albastru, cu valoarea P_gri obtinuta
- Trebuie sa modificati imaginea inițială fără a o salva într-un fișier nou.

Basic BMP File Format			
Name		**Size**	**Description**
Header		14 bytes	Windows Structure: BITMAPFILEHEADER
	Signature	2 bytes	'BM'
	FileSize	4 bytes	File size in bytes
	reserved	4 bytes	unused (=0)
	DataOffset	4 bytes	File offset to Raster Data
InfoHeader		40 bytes	Windows Structure: BITMAPINFOHEADER
	Size	4 bytes	Size of InfoHeader =40
	Width	4 bytes	Bitmap Width
	Height	4 bytes	Bitmap Height
	Planes	2 bytes	Number of Planes (=1)
	BitCount	2 bytes	Bits per Pixel 1 = monochrome palette. NumColors = 1 4 = 4bit palletized. NumColors = 16 8 = 8bit palletized. NumColors = 256 16 = 16bit RGB. NumColors = 65536 (?) 24 = 24bit RGB. NumColors = 16M
	Compression	4 bytes	Type of Compression 0 = BI_RGB no compression 1 = BI_RLE8 8bit RLE encoding 2 = BI_RLE4 4bit RLE encoding
	ImageSize	4 bytes	(compressed) Size of Image It is valid to set this =0 if Compression = 0
	XpixelsPerM	4 bytes	horizontal resolution: Pixels/meter
	YpixelsPerM	4 bytes	vertical resolution: Pixels/meter
	ColorsUsed	4 bytes	Number of actually used colors
	ColorsImportant	4 bytes	Number of important colors 0 = all
ColorTable		4 * NumColors bytes	present only if Info.BitsPerPixel <= 8 colors should be ordered by importance
	Red	1 byte	Red intensity
	Green	1 byte	Green intensity
	Blue	1 byte	Blue intensity
	reserved	1 byte	unused (=0)
	repeated NumColors times		
Raster Data		Info.ImageSize bytes	The pixel data