

---

# **CSC153: Activity 6: Recovering Graphics Files**

Ryan Kozak



2019-11-10

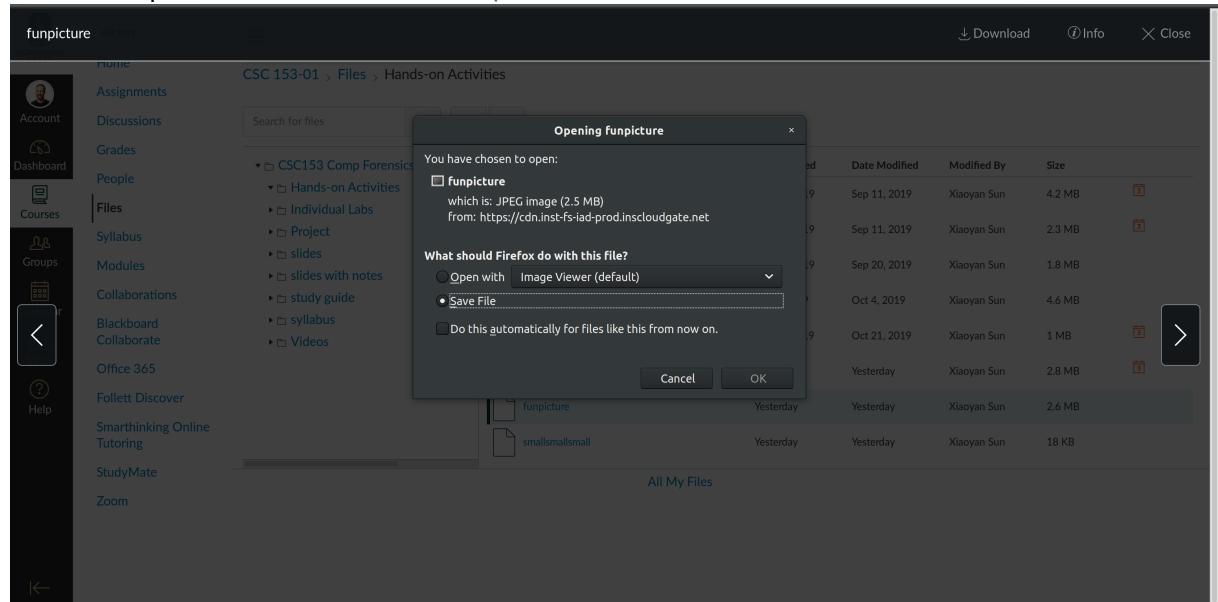
## Objectives

- Split and combine files in Linux.
- Use WinHex to recover graphics files.

### Part 1: Split and combine files in Linux.

Since evaluation version of Winhex cannot process files larger than 200KBs, we must split the file into pieces in order to edit it.

The first step is to download the file [funpicture](#) from Canvas onto our local Linux machine.



**Figure 1:** Downloading [funpicture](#) to our local machine.

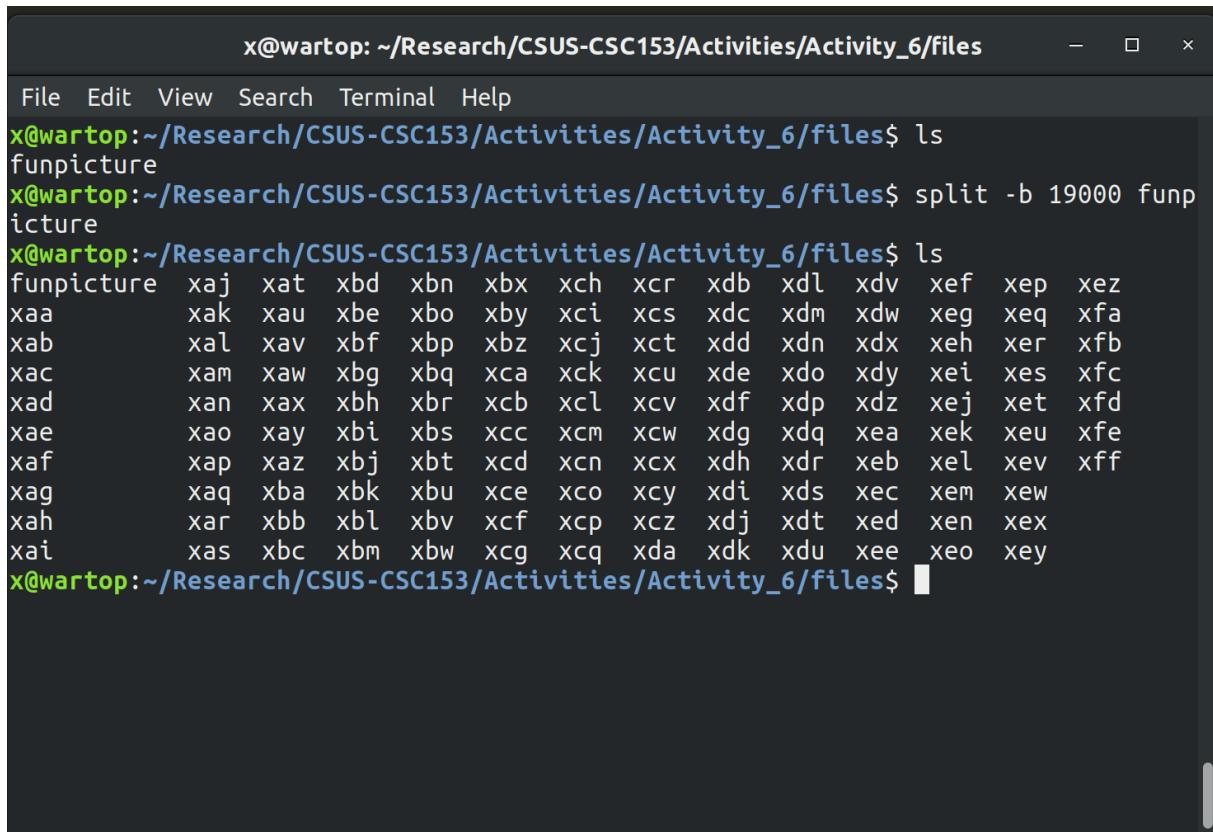
We then open the terminal and navigate to the directory that contains this file.



```
x@wartop: ~/Research/CSUS-CSC153/Activities/Activity_6/files
File Edit View Search Terminal Help
x@wartop:~$ cd ~/Research/CSUS-CSC153/Activities/Activity_6/files
x@wartop:~/Research/CSUS-CSC153/Activities/Activity_6/files$ 
```

**Figure 2:** Change directory to where we've downloaded `funpicture`.

Now we must split the file via the `split -b 19000 funpicture`.



The screenshot shows a terminal window with the following session:

```
x@wartop: ~/Research/CSUS-CSC153/Activities/Activity_6/files
File Edit View Search Terminal Help
x@wartop:~/Research/CSUS-CSC153/Activities/Activity_6/files$ ls
funpicture
x@wartop:~/Research/CSUS-CSC153/Activities/Activity_6/files$ split -b 19000 funpicture
x@wartop:~/Research/CSUS-CSC153/Activities/Activity_6/files$ ls
funpicture  xaj  xat  xbd  xbn  xbx  xch  xcr  xdb  xdl  xdv  xef  xep  xez
xaa        xak  xau  xbe  xbo  xby  xci  xcs  xdc  xdm  xdw  xeg  xeq  xfa
xab        xal  xav  xbf  xbp  xbz  xcj  xct  xdd  xdn  xdx  xeh  xer  xfb
xac        xam  xaw  xbg  xbg  xca  xck  xcu  xde  xdo  xdy  xei  xes  xfc
xad        xan  xax  xbh  xbr  xcb  xcl  xcv  xdf  xdp  xdz  xej  xet  xfd
xae        xao  xay  xbi  xbs  xcc  xcm  xcw  xdg  xdq  xea  xek  xeu  xfe
xaf        xap  xaz  xbj  xbt  xcd  xcn  xcx  xdh  xdr  xeb  xel  xev  xff
xag        xaq  xba  xbk  xbu  xce  xco  xcy  xdi  xds  xec  xem  xew
xah        xar  xbb  xbl  xbv  xcf  xcp  xcz  xdj  xdt  xed  xen  xex
xai        xas  xbc  xbm  xbw  xcg  xcq  xda  xdk  xdu  xee  xeo  xey
x@wartop:~/Research/CSUS-CSC153/Activities/Activity_6/files$
```

**Figure 3:** Splitting `funpicture` file, and listing results.

After splitting the file we recombine it with the command `cat x*>newfun`.

The screenshot shows a terminal window with a dark background and light-colored text. At the top, the title bar reads "x@wartop: ~/Research/CSUS-CSC153/Activities/Activity\_6/files". The menu bar includes "File", "Edit", "View", "Search", "Terminal", and "Help". Below the menu, the terminal prompt is "x@wartop:~/Research/CSUS-CSC153/Activities/Activity\_6/files\$". The user has run two commands: "cat x\*>newfun" and "ls". The output of "cat" shows a large block of text where each line consists of two groups of four lowercase letters separated by a space. The output of "ls" lists 16 files, each corresponding to one of the lines produced by the "cat" command. The files are: funpicture, newfun, xaa, xab, xac, xad, xae, xaf, xag, xah.

**Figure 4:** Recombining the image with the `cat` command.

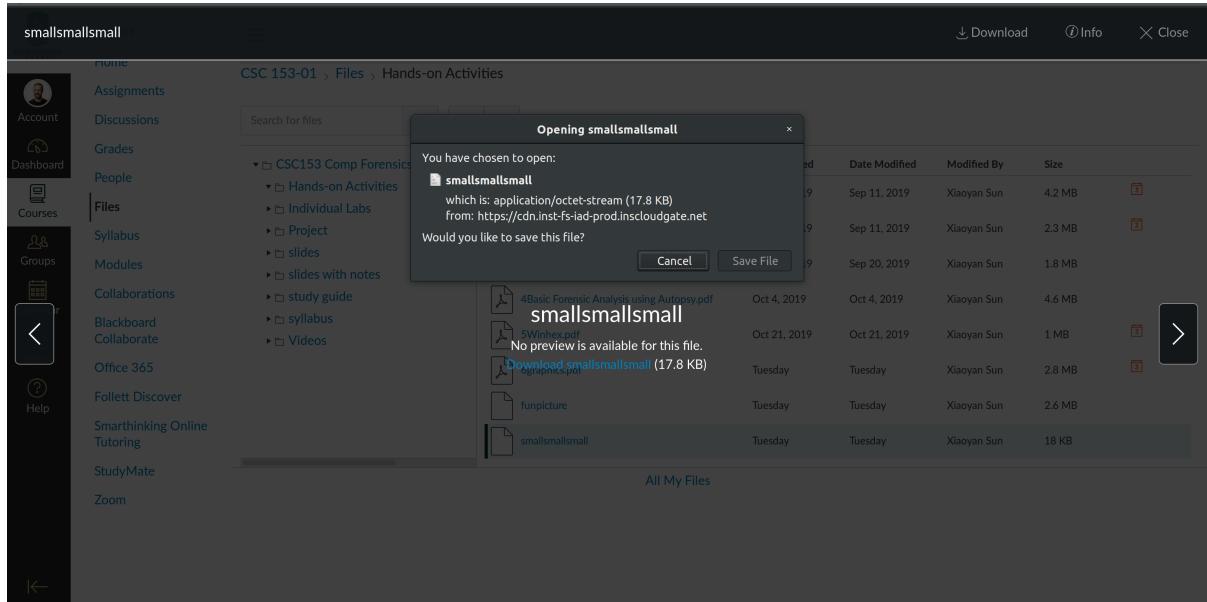
The last step for part 1 is to rename the combined file to include the `.jpg` extension, via `mv newfile newfile.jpg`.

```
x@wartop:~/Research/CSUS-CSC153/Activities/Activity_6/files$ mv newfun newfun.jpg
x@wartop:~/Research/CSUS-CSC153/Activities/Activity_6/files$ ls
funpicture  xai  xas  xbc  xbm  xbw  xcg  xcq  xda  xdk  xdu  xee  xeo  xey
newfun.jpg  xaj  xat  xbd  xbn  xbx  xch  xcr  xdb  xdl  xdv  xef  xep  xez
xaa        xak  xau  xbe  xbo  xby  xci  xcs  xdc  xdm  xdw  xeg  xeq  xfa
xab        xal  xav  xbf  xbp  xbz  xcj  xct  xdd  xdn  xdx  xeh  xer  xfb
xac        xam  xaw  xbg  xbg  xca  xck  xcu  xde  xdo  xdy  xei  xes  xfc
xad        xan  xax  xbh  xbr  xcb  xcl  xcv  xdf  xdp  xdz  xej  xet  xfd
xae        xao  xay  xbi  xbs  xcc  xcm  xcw  xdg  xdq  xea  xek  xeu  xfe
xaf        xap  xaz  xbj  xbt  xcd  xcn  xcx  xdh  xdr  xeb  xel  xev  xff
xag        xaq  xba  xbk  xbu  xce  xco  xcy  xdi  xds  xec  xem  xew
xah        xar  xbb  xbl  xbv  xcf  xcp  xcz  xdj  xdt  xed  xen  xex
x@wartop:~/Research/CSUS-CSC153/Activities/Activity_6/files$
```

**Figure 5:** Rename `newfile` to `newfile.jpg`.

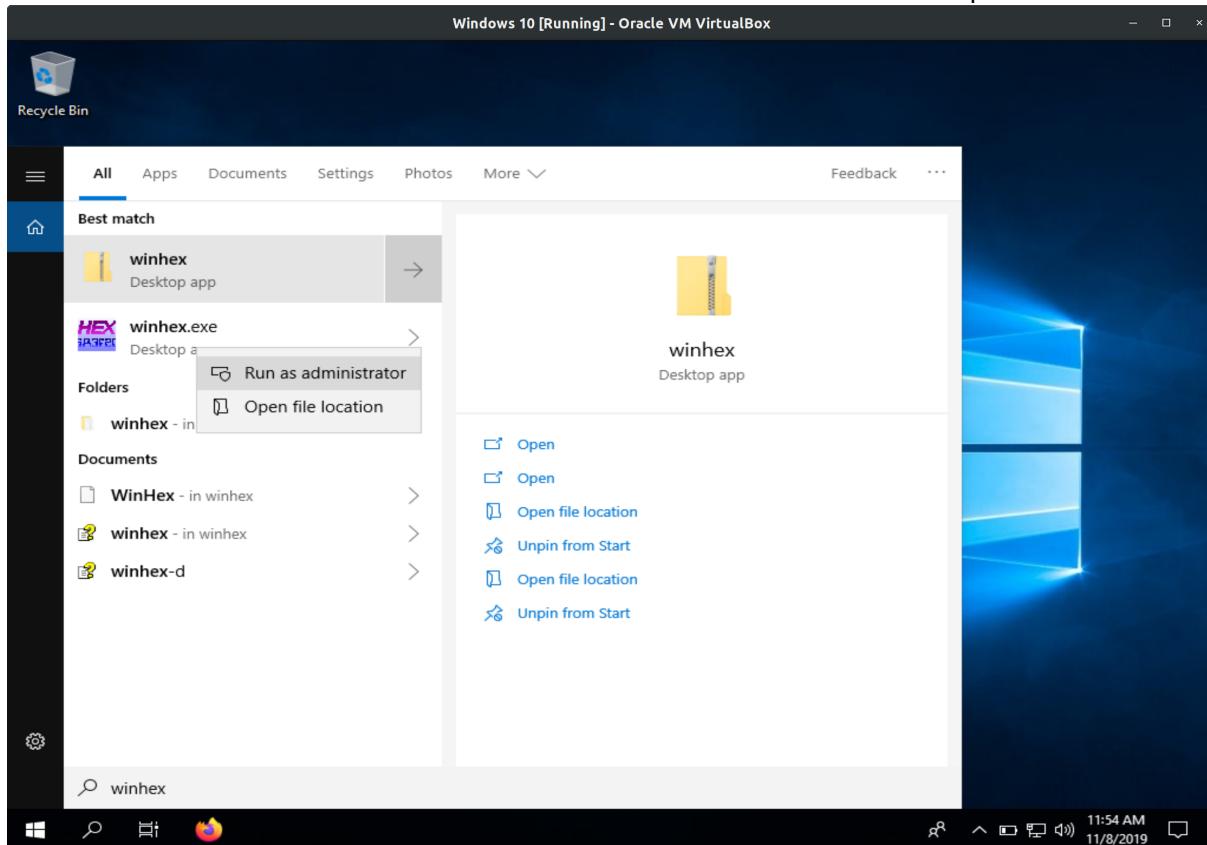
## Part 2: Practice recovering graphics files using Winhex.

Now we download the file `smallsmallsmall` from Canvas and save it to our working directory folder. This file is a [PNG](#) file, but the header has been modified by the suspect.



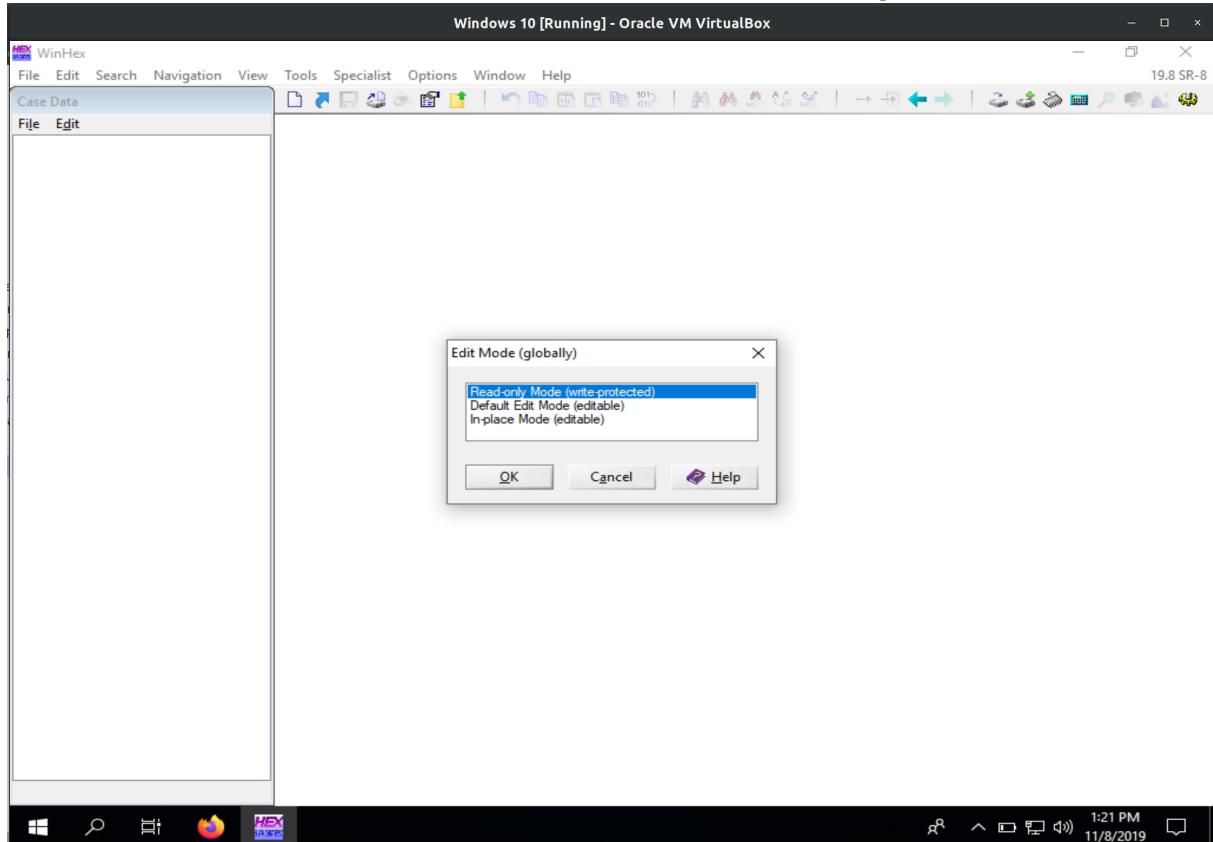
**Figure 6:** Downloading `smallsmallsmall` to our local machine, in a shared folder with our the Windows VM.

Now we boot our Windows VM and start WinHex with the Run as administrator option.

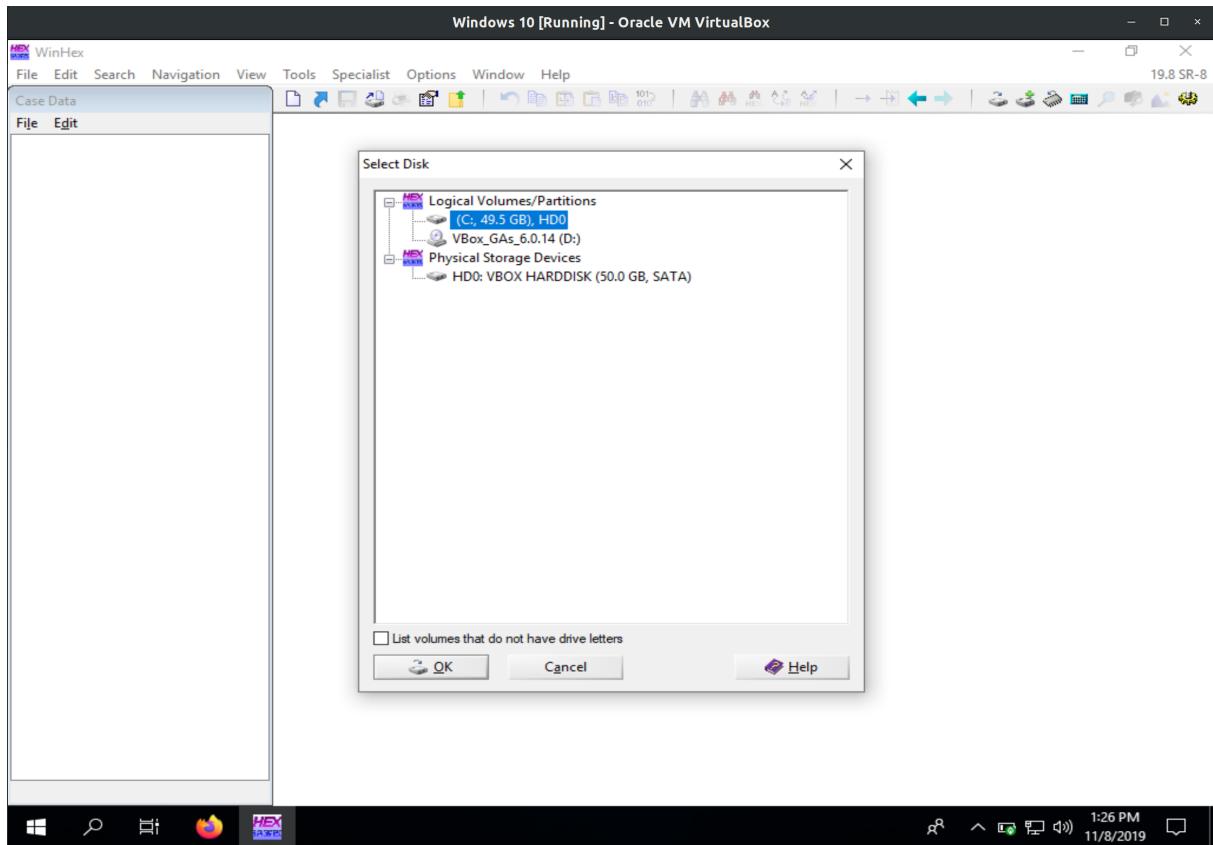


**Figure 7:** Launching WinHex as Admin.

As a safety precaution, we click **Options** → **Edit Mode** from the menu. In the Select Mode dialog box, we click **Read-Only Mode (=write protected)**, as shown in Figure 8, and then click **OK**.

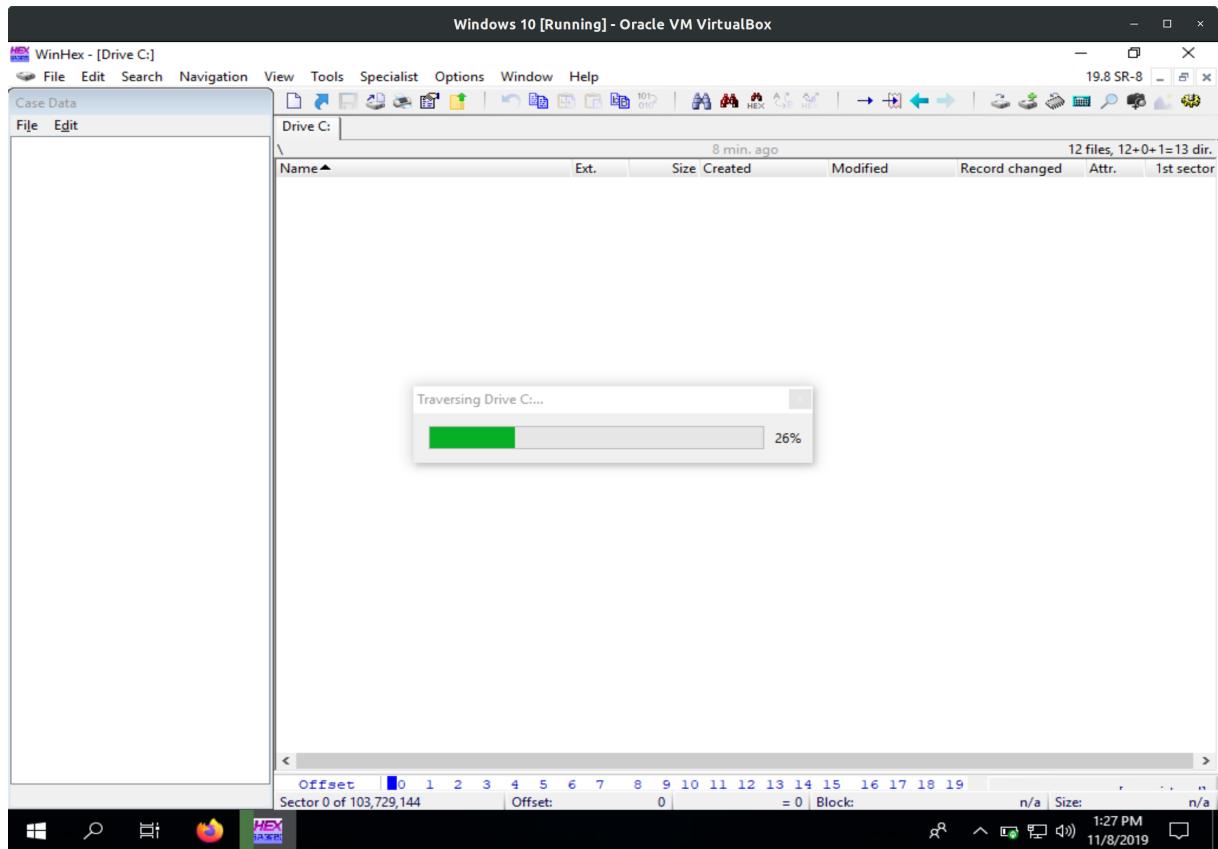
**Figure 8:** Select read-only mode for safety.

Next we click **Tools** → **Open Disk** from the menu. In the View Disk dialog box, we click the drive where we saved **smallsmallsmall**. In our case we've moved the file to the **Desktop**, so we'll select the **C:** drive.



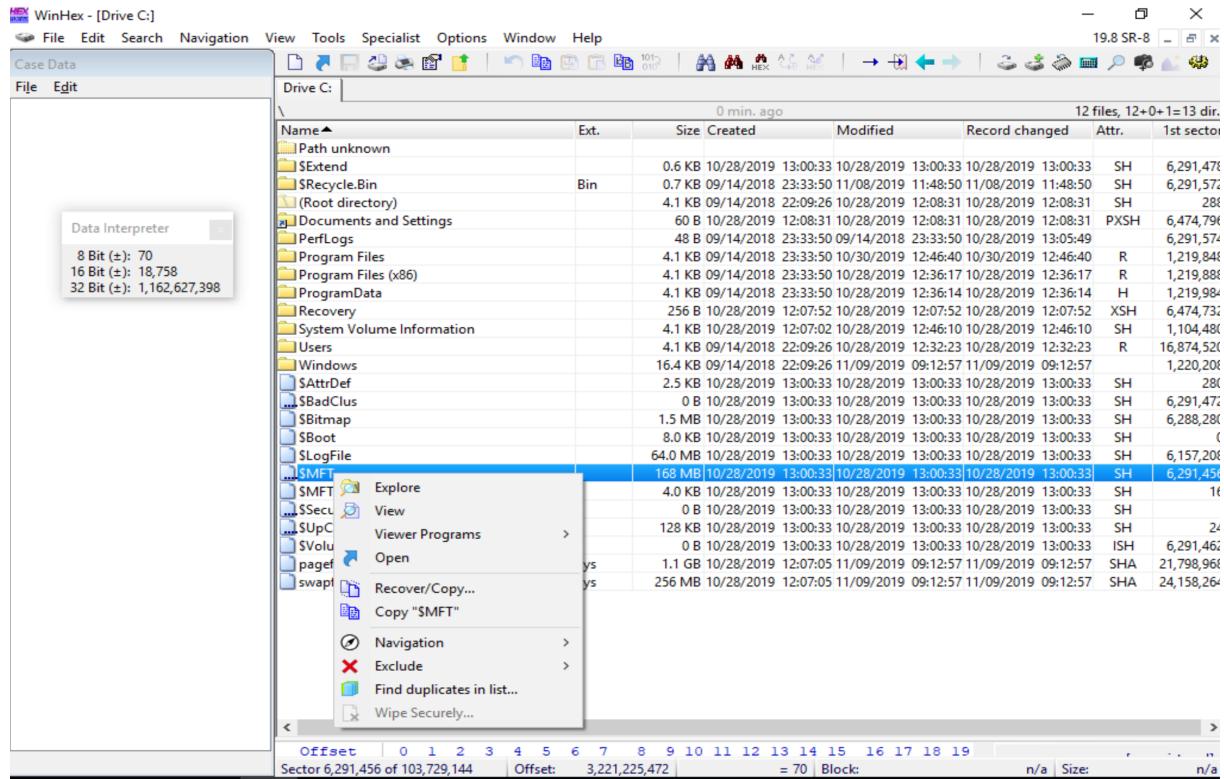
**Figure 9:** Selecting the C: drive to open and examine.

After we click **Ok** WinHex begins traversing the C: drive.



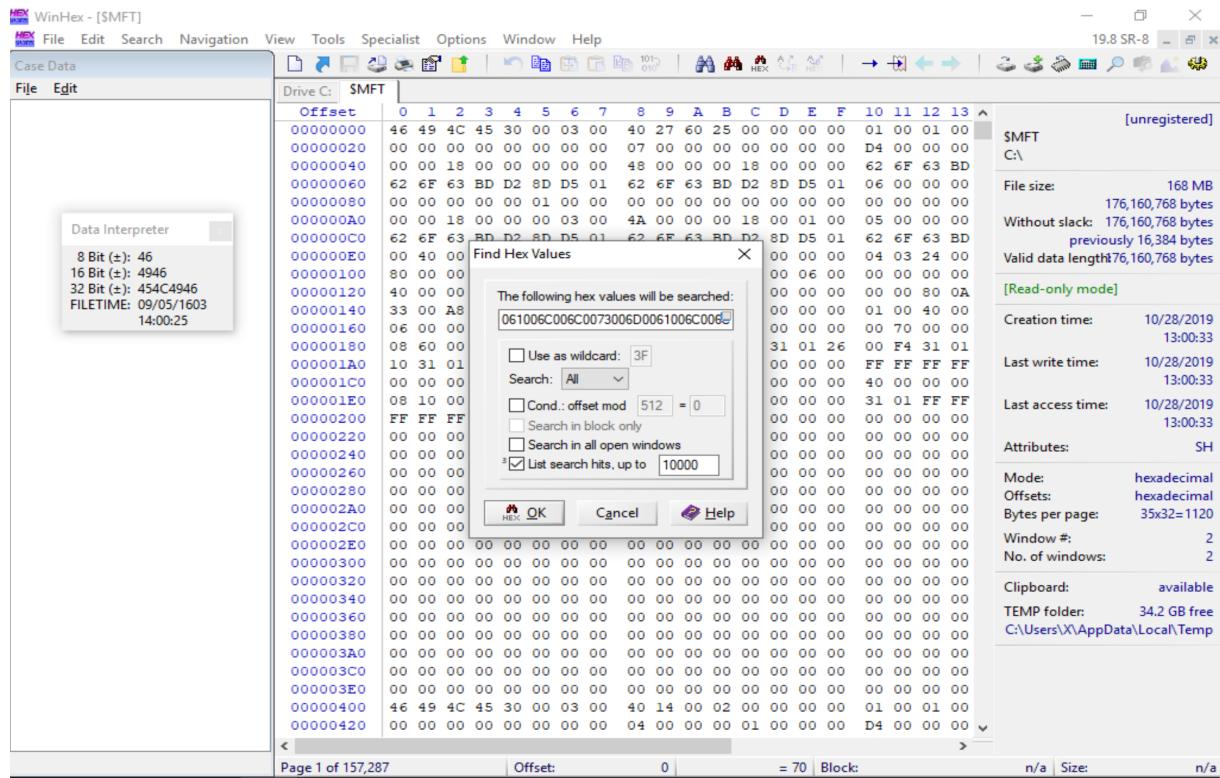
**Figure 10:** WinHex traversing our C: drive.

Now we scroll down and find the \$MFT file, right click and choose open. We open the MFT file in a new window as seen in the figure 11 below.



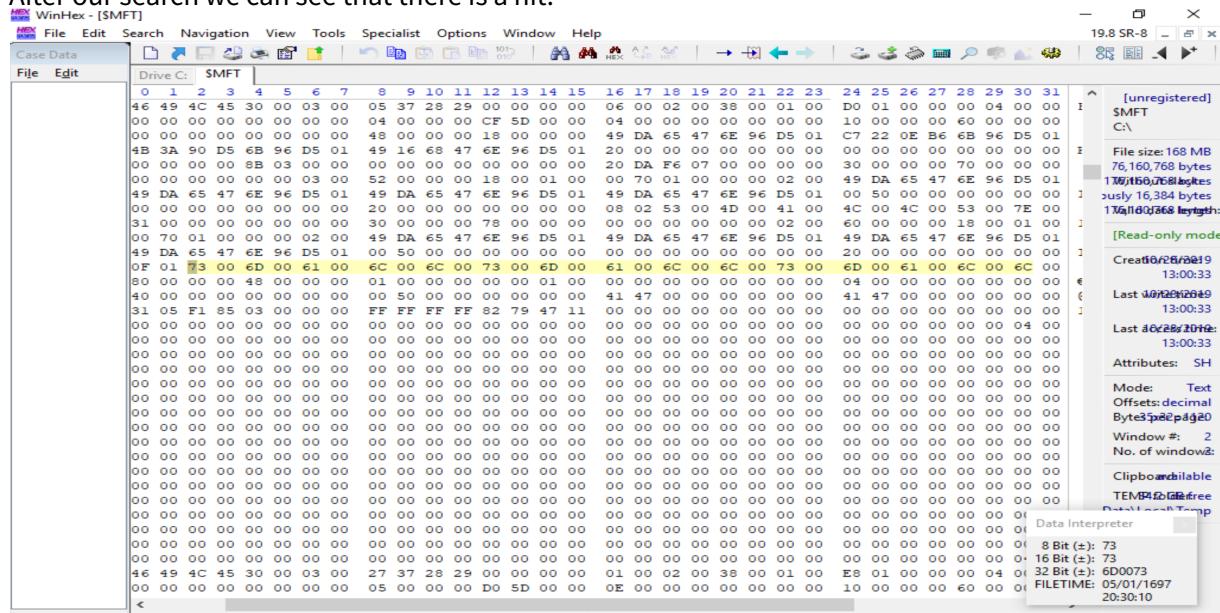
**Figure 11:** Opening \$MFT to search for our file.

We click Search → Find Hex Values, and query the hexadecimal string we calculated above.



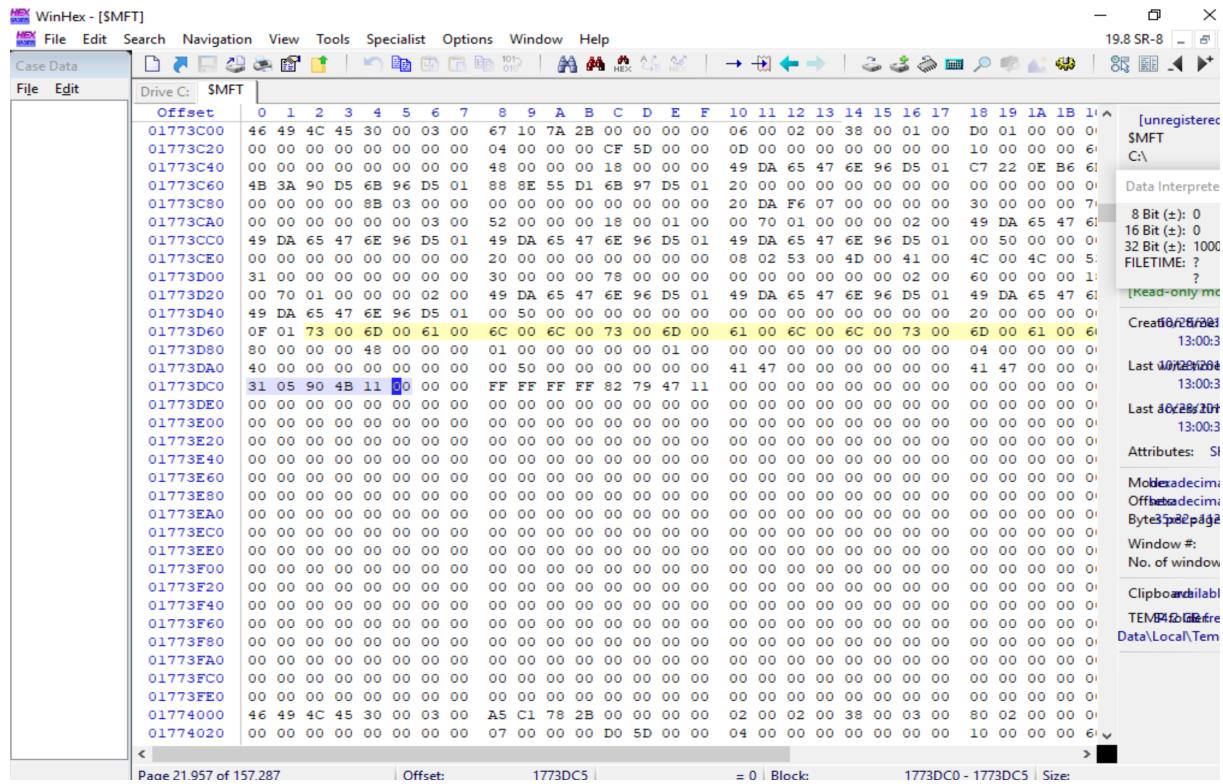
**Figure 12:** Searching for the hexadecimal title of smallsmallsmall.

After our search we can see that there is a hit.



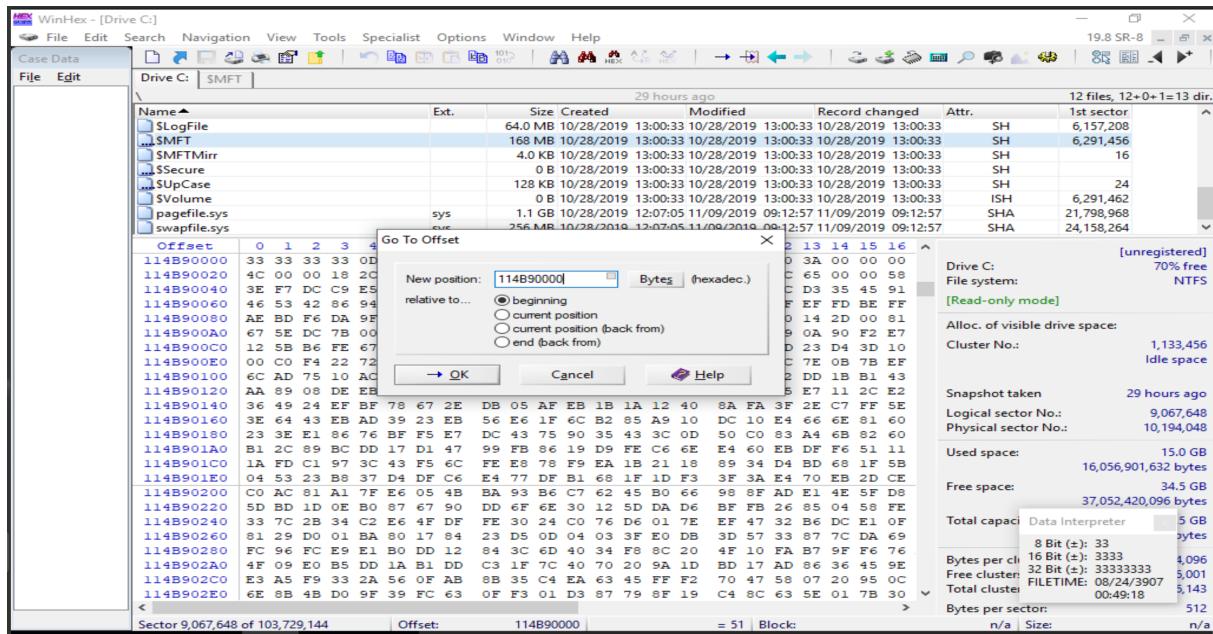
**Figure 13:** A hit for the hexadecimal search.

Following the same methodology as we did in hands-on activity 5 , we examine the \$MFT file record. We can find the start of the data run for this file by looking at offset 0x40 from attribute 0x80.



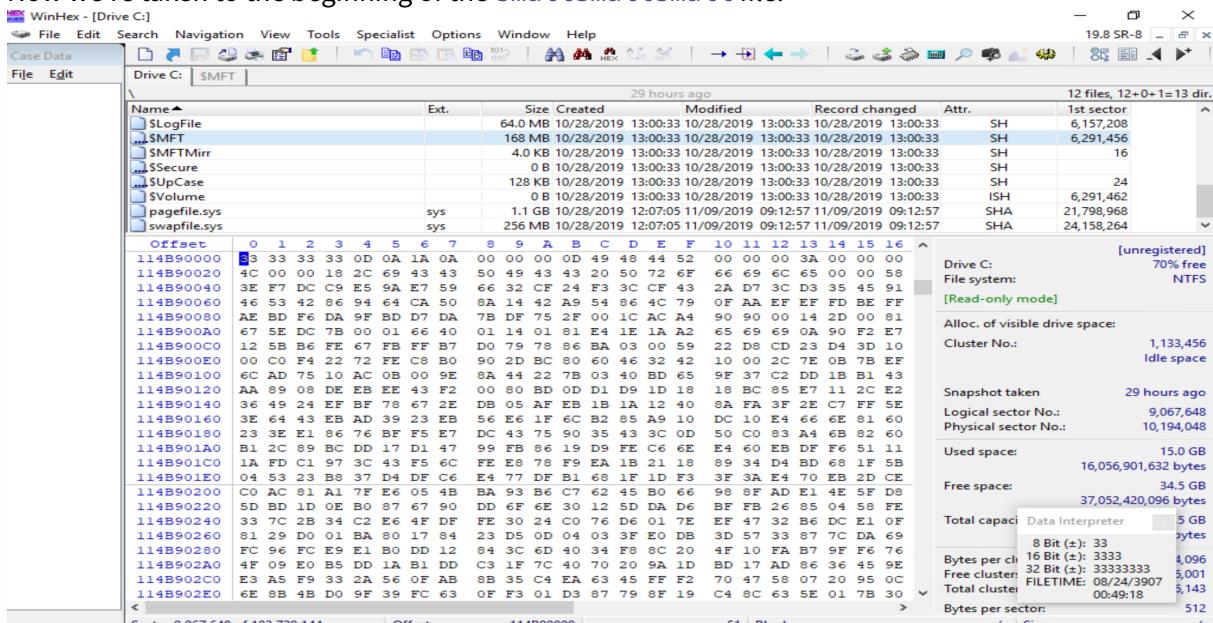
**Figure 14:** First data run info is 31 05 90 4B 11 00.

The first data run is 31 05 90 4B 11 00, which means the starting position for the first data run is `0x 00 11 4B 90` and the size is `0x05`. So next, we open the window for the C: drive once again, and click on `Navigation -> Go To Offset`. We'll go to the offset we've just found, appending three zeros to the end of it (see Figure 15 below).



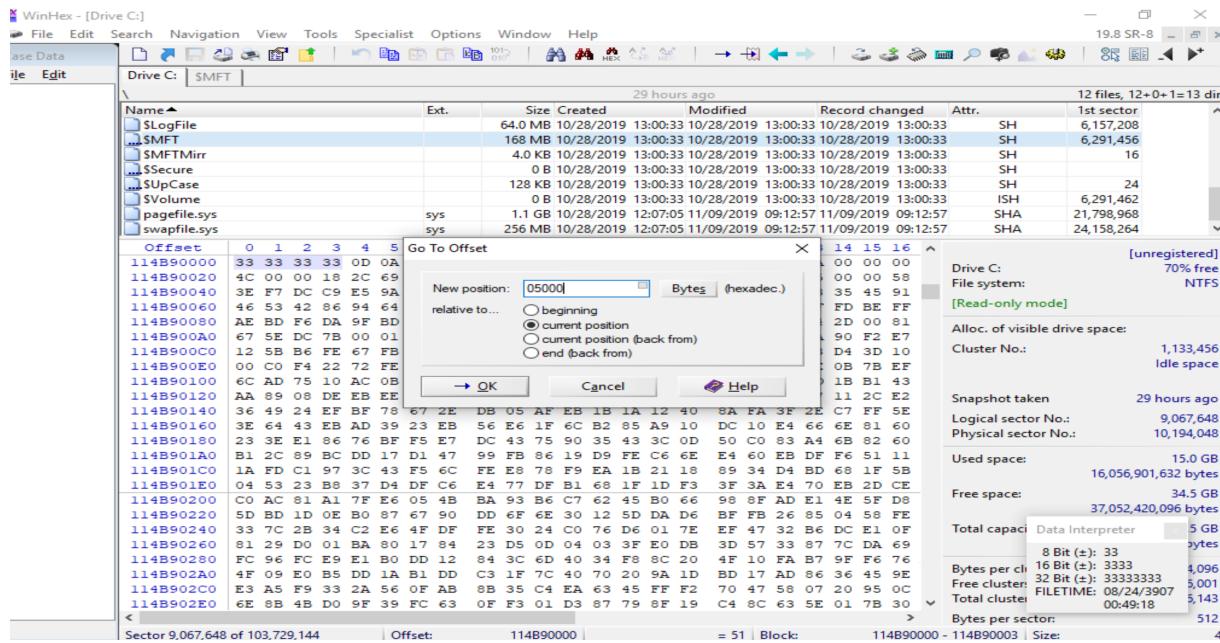
**Figure 15:** The first data run at offset 114B90000.

Now we're taken to the beginning of the `smallsmallsmall` file.



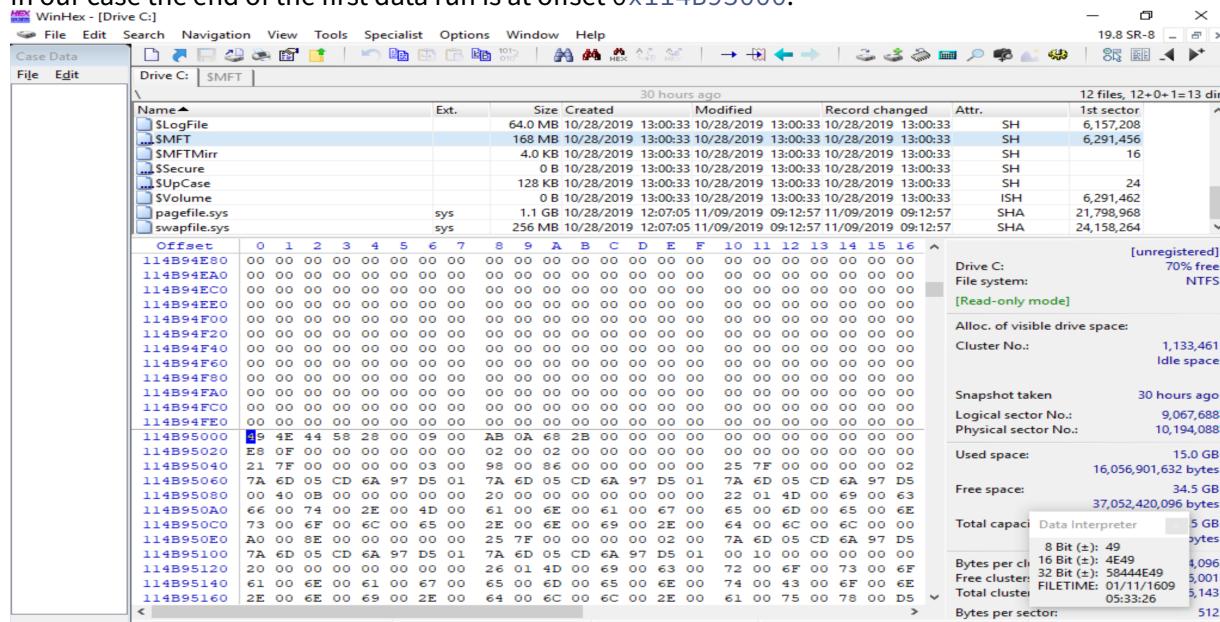
**Figure 16:** Beginning of `smallsmallsmall` file.

Since the size of the first data run is `0x05`, we can click on **Navigation** -> **Go To Offset** again to find the end of the first data run. Setting `05000` as our position and choosing **current position** will take us to the ending point for the first data run.



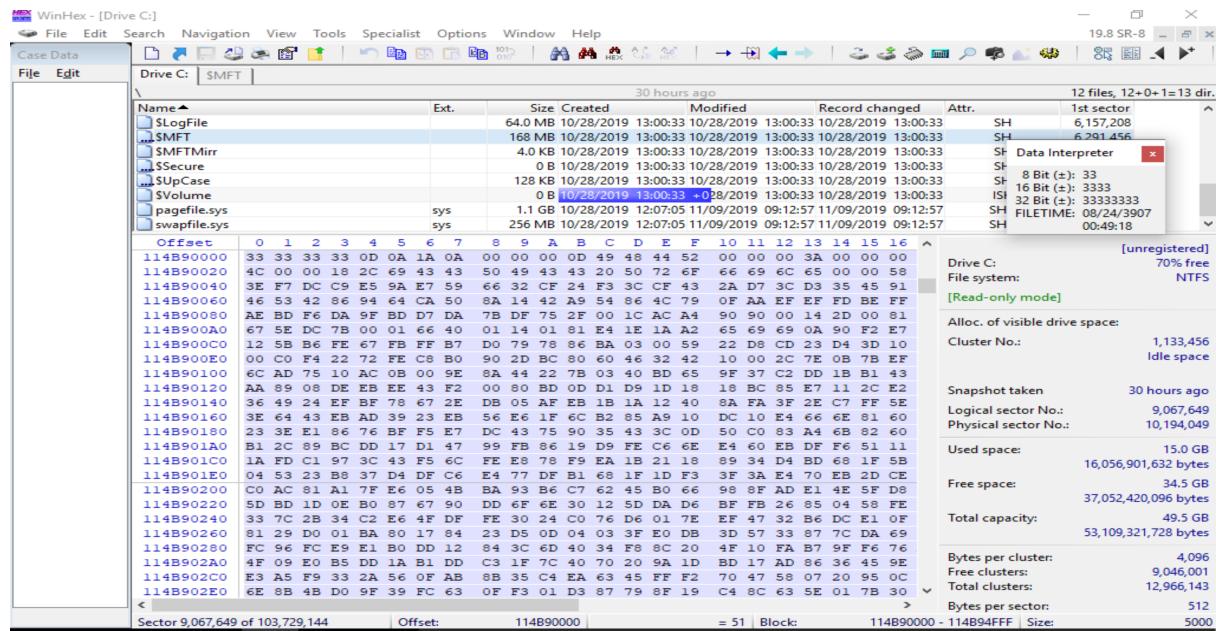
**Figure 17:** Go To Offset at the end of first data run.

In our case the end of the first data run is at offset **0x114B95000**.

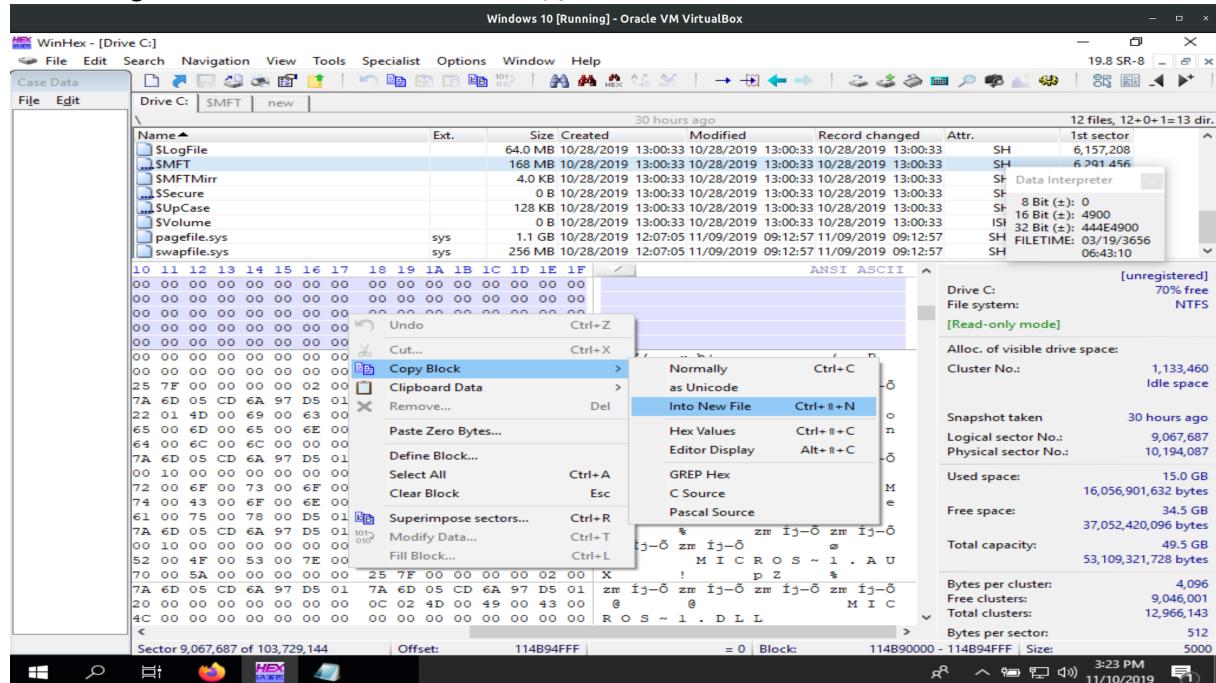


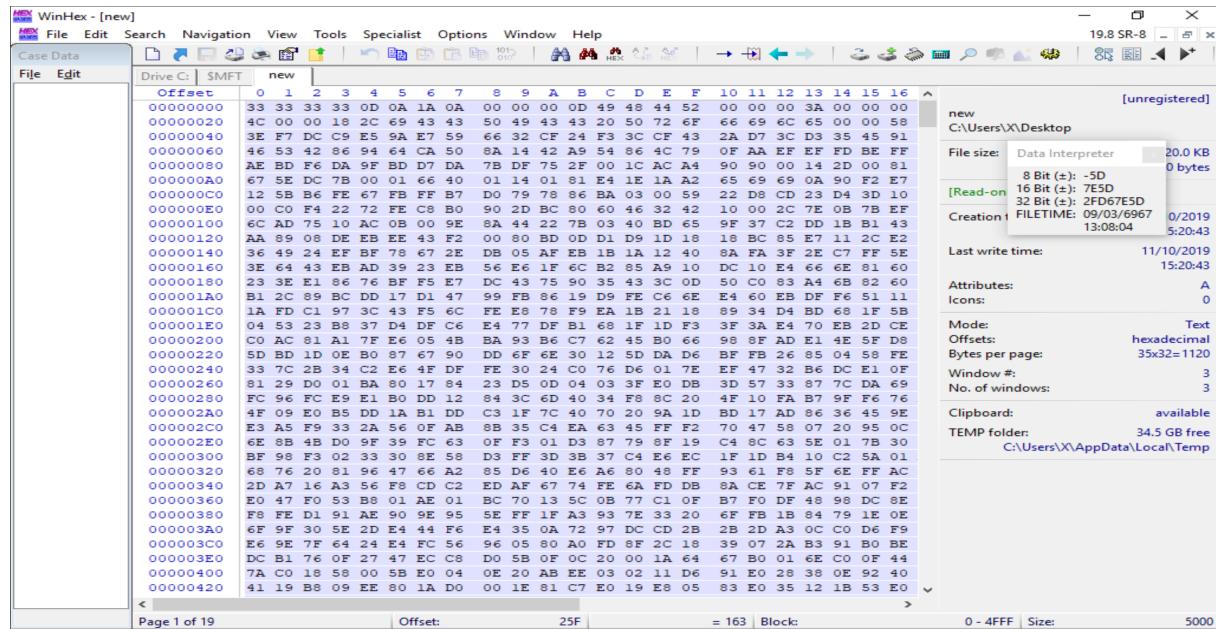
**Figure 18:** End of the first data run at.

Starting from the beginning of the file again, we click on the first byte of the file and drag it until the offset is 5000, which is the size of the data run.

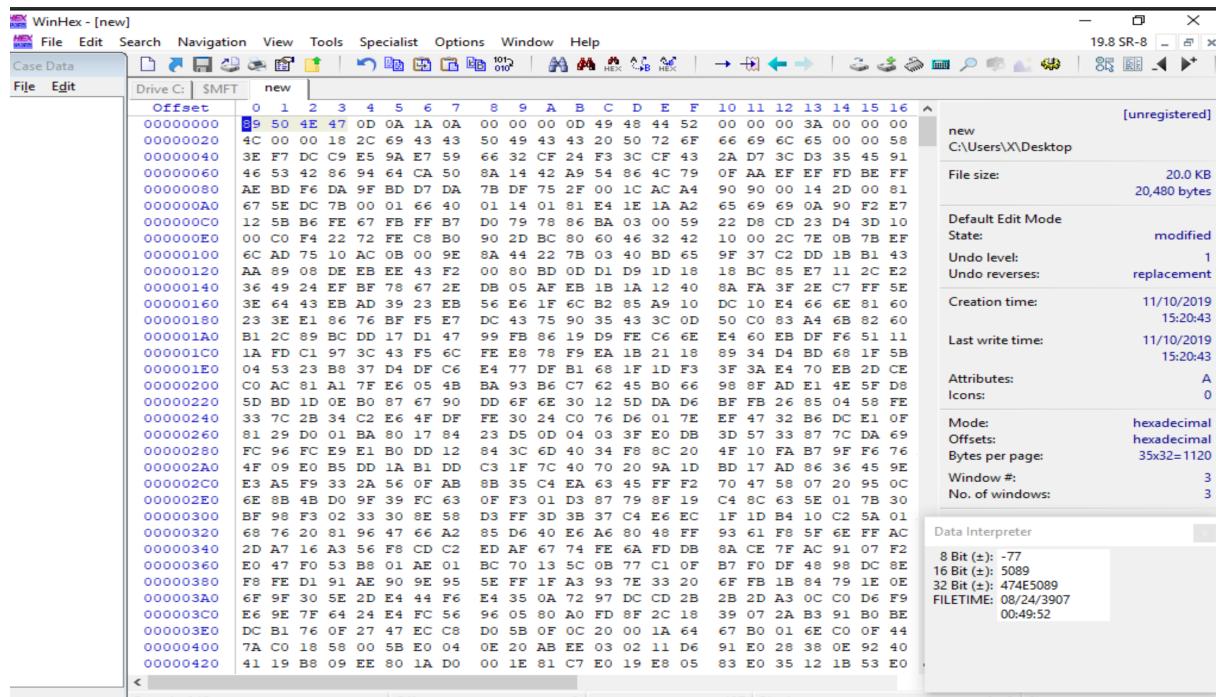
**Figure 19:** Selecting everything from first data run.

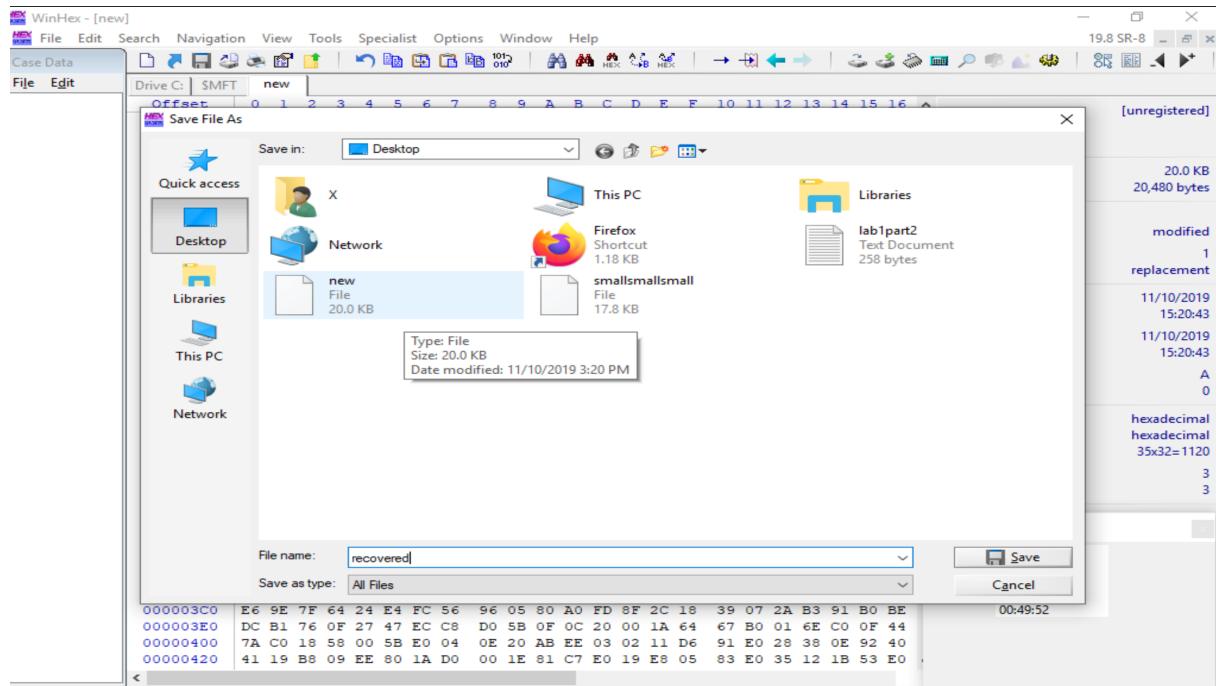
Next we right click and choose **Edit** → **Copy Block** → **Into New File**.

**Figure 20:** Copy block to new file.

**Figure 21:** New file from data run copied in figure 20.

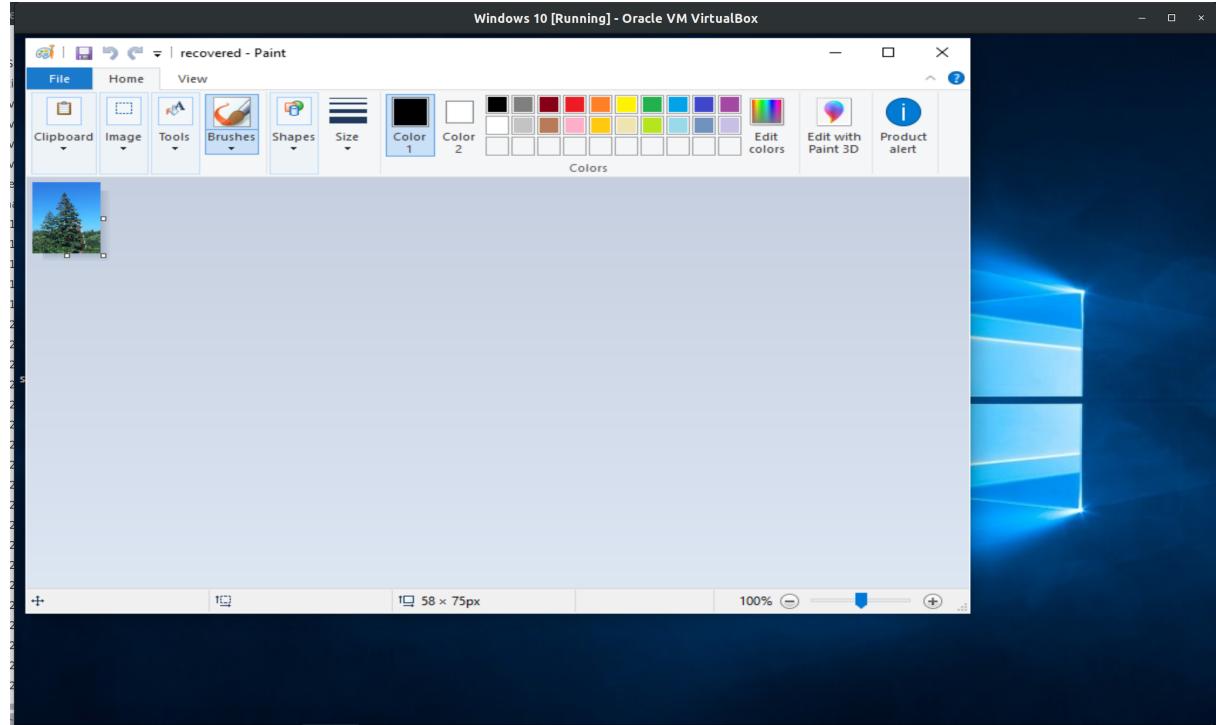
The correct header for the PNG file format is 89 50 4E 47, so we change the header of the new file from 33 33 33 33 to 89 50 4E 47, and save it as recovered.

**Figure 22:** Replace file header with proper header for PNG.



**Figure 23:** Saving file as *recovered*.

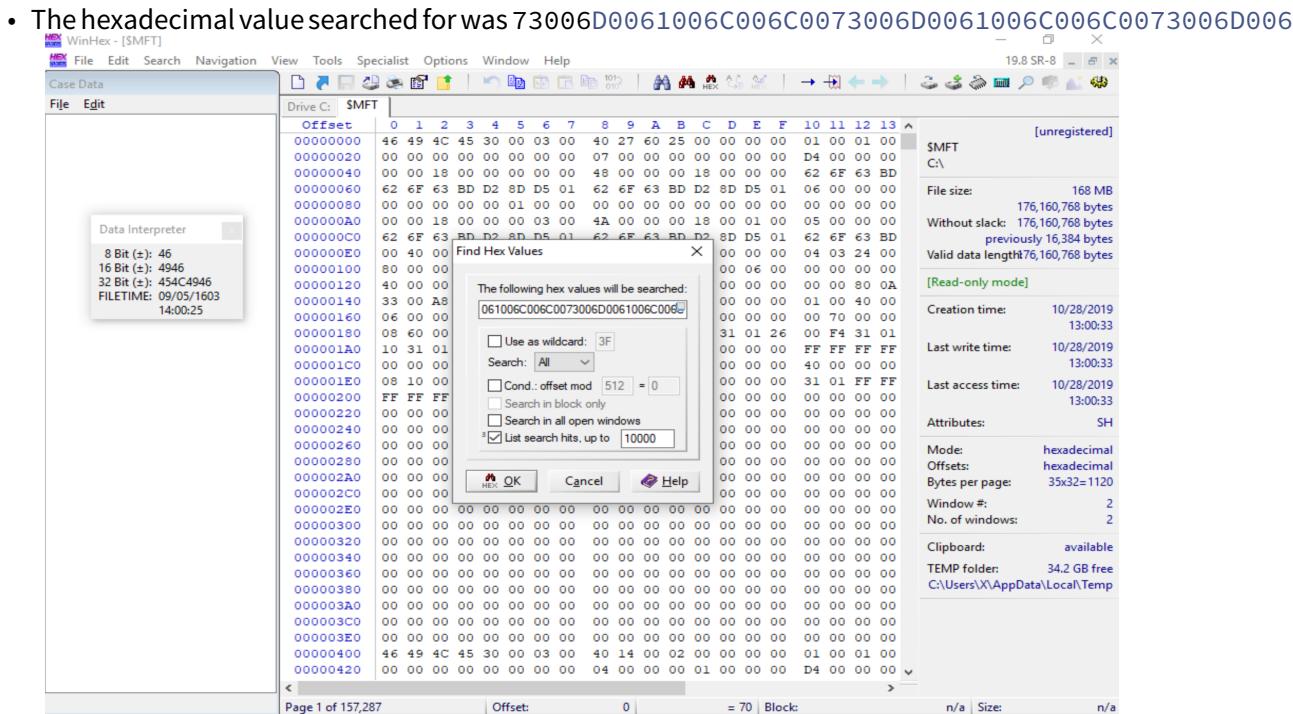
Now once we add the .png extension, we can open the file `recovered.png` and see that it's a 58x75 pixel version of the image from Part 1.



**Figure 24:** Opening recovered file.

## Questions

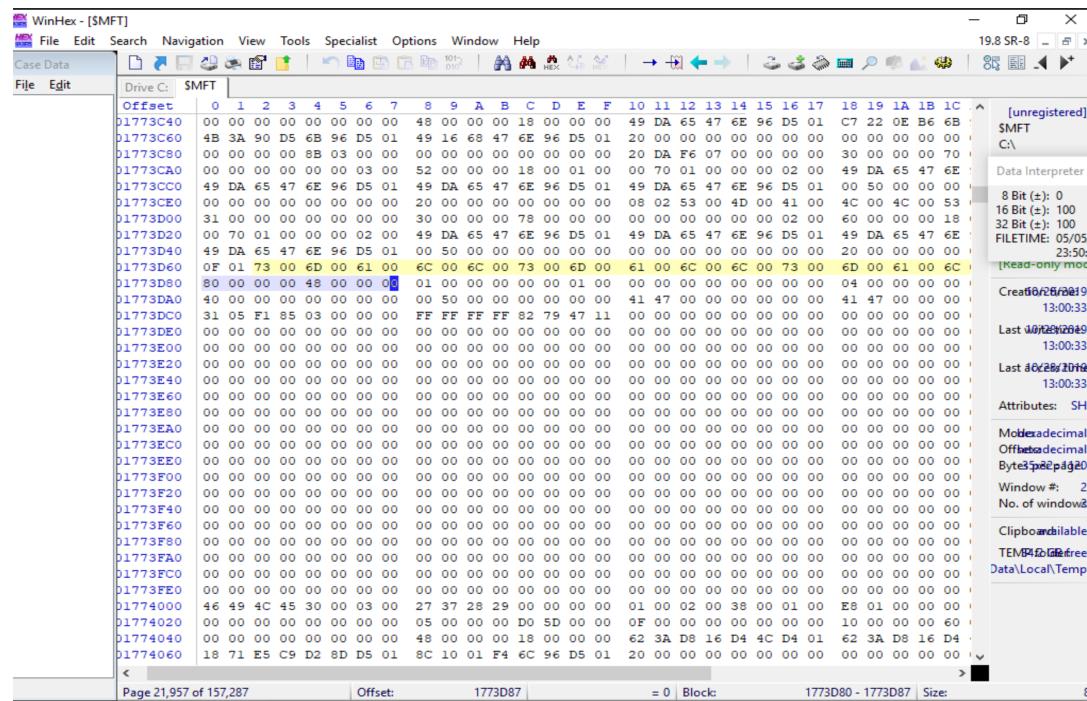
- What were the hexadecimal values did you used to search for the file `smallsmallsmall` in steps 13 and 15?



**Figure 25:** Searching for the hexadecimal title of `smallsmallsmall`.

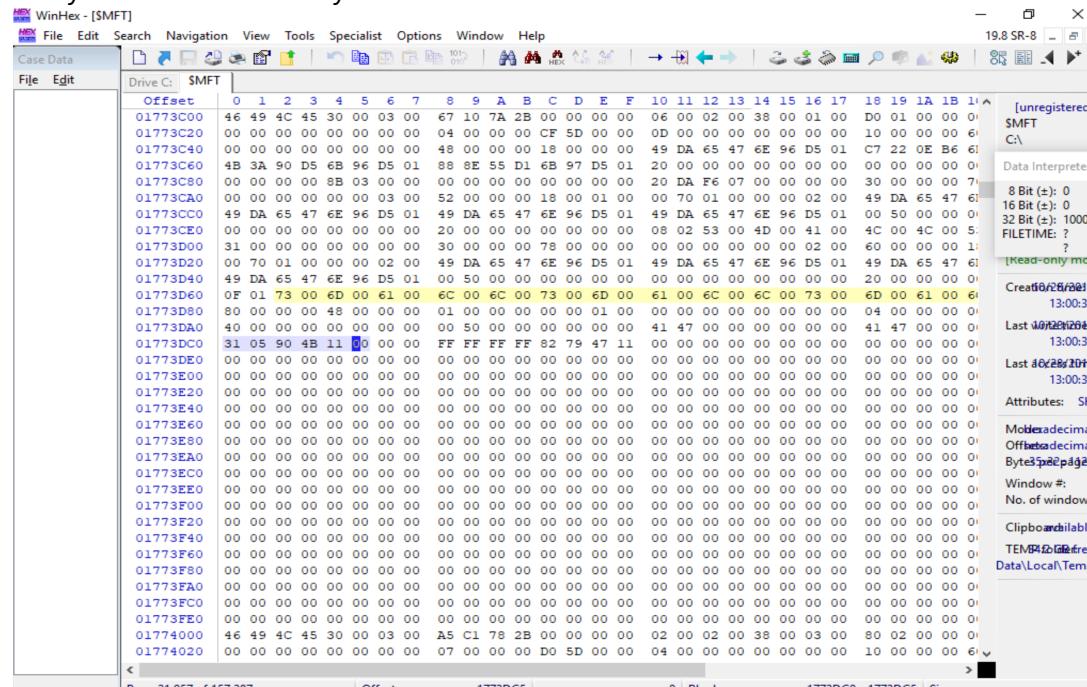
- Is the file a resident file or non-resident file? How do you know?

- This is a **non-resident** file, we can tell by looking at offset `0x08` from attribute `0x80` and see the flag is set to `0x01`.

**Figure 26:** Non-resident flag.

### 3. How many data runs does this file has?

- In my case this file had only one data run.

**Figure 27:** One data run for this file.

4. What is the starting position for the first data run?

- The starting position for the first data run is at offset `0x00114B90`.

The screenshot shows the WinHex application window. The menu bar includes File, Edit, Search, Navigation, View, Tools, Specialist, Options, Window, and Help. The toolbar contains various icons for file operations like Open, Save, Copy, Paste, and Find. The status bar at the bottom shows page numbers (Page 21,957 of 157,287), offsets (Offset: 1773DC5), and block sizes (Size: 1773DC0 - 1773DC5). The main area displays a hex dump of memory starting at offset 0. The data is presented in three columns: Address (Offset), Hex, and ASCII. The address column shows values from 0 to 1774020. The hex column shows binary data in groups of four bytes. The ASCII column shows the corresponding characters or symbols. A vertical scroll bar is visible on the right side of the data pane.

**Figure 28:** Starting position for first data run.

5. What is the size of the first data run?

- The size of the first data run is `0x05`.

The screenshot shows the WinHex application window. The menu bar includes File, Edit, Search, Navigation, View, Tools, Specialist, Options, Window, Help, and a 1010 icon. The toolbar contains icons for Open, Save, Find, Replace, Copy, Paste, Cut, Delete, Insert, Find Next, Find Previous, and various selection and search tools. The status bar at the bottom displays "Page 21 957 of 157 287", "Offset: 1773DC5", "= 0 | Block:", "1773DC0 - 1773DC5 | Size:", and "19.8 SR-8". The main window displays a hex dump of the file SMFT. The left column shows the offset from 0 to 1000. The right column shows the ASCII representation of the data. A vertical scroll bar is visible on the right side of the data pane.

**Figure 29:** Size of first data run is 0x05.

6. In step 20, what is the header of the file `smallsmallsmall` which you downloaded from Canvas? Please provide the first 4 bytes of the hexadecimal values.

- The header of the file downloaded from Canvas was 33 33 33 33.

Name	Ext.	Size	Created	Modified	Record changed	Attr.	1st sector
SMFT							
SLogFile		64.0 MB	10/28/2019	13:00:33	10/28/2019	13:00:33	10/28/2019
...\$MFT		168 MB	10/28/2019	13:00:33	10/28/2019	13:00:33	10/28/2019
...\$MFTMirr		4.0 KB	10/28/2019	13:00:33	10/28/2019	13:00:33	10/28/2019
...\$Secure		0 B	10/28/2019	13:00:33	10/28/2019	13:00:33	10/28/2019
...\$UpCase		12 KB	10/28/2019	13:00:33	+08/2019	13:00:33	10/28/2019
SVolume		0 B	10/28/2019	13:00:33	10/28/2019	13:00:33	10/28/2019
pagefile.sys	sys	1.1 GB	10/28/2019	12:07:05	11/09/2019	09:12:57	10/28/2019
swapfile.sys	sys	256 MB	10/28/2019	12:07:05	11/09/2019	09:12:57	10/28/2019

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
114B900000	33	33	33	33	33	0D	0A	1A	0A	00	00	0D	49	48	44	52	00	00	3A	00	00	00	00
114B900040	4C	00	00	1A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
114B900400	3E	DC	89	E5	5A	E7	59	66	00	00	00	24	FC	0C	CF	8	0A	D7	3C	D3	39	45	FF
114B900460	46	53	42	86	94	64	CA	50	8A	14	42	A9	54	86	4C	79	0F	AA	EF	EF	FD	BE	FF
114B900800	AE	BD	F6	DA	9F	BD	D7	DA	75	DI	75	2F	00	1C	AC	A4	90	90	00	14	2D	00	81
114B900A00	67	5E	DC	7B	00	01	66	40	01	14	01	81	E4	1E	1A	A2	65	69	60	0A	90	F2	E7
114B900C00	12	5B	B6	FE	67	FB	FF	B7	D0	75	78	86	BA	03	00	59	22	D0	C0	23	D4	3D	10
114B900E00	00	00	F4	22	72	FE	C8	B0	90	2D	BC	80	60	46	32	42	10	00	2C	7E	0B	7B	EF
114B901000	6C	AD	75	10	AC	0B	00	9E	8A	44	22	7B	03	40	BD	65	9F	37	C2	DD	1B	B1	43
114B901200	AA	89	08	0D	EE	EE	43	F2	00	80	BD	0D	D1	D9	1D	18	BC	85	E7	11	2C	E2	2E
114B901400	36	49	24	EF	BF	78	67	2E	DD	05	AF	EB	1B	1A	12	40	8A	FA	3F	2E	C7	FF	SE
114B901600	3E	64	43	EB	AD	39	23	EB	56	E6	1F	EC	B2	85	A9	10	DC	10	E4	66	6E	81	60
114B901800	23	3E	E1	86	70	BF	F5	E7	DC	43	75	90	35	43	3C	0D	50	C0	83	A4	6B	62	60
114B901A00	B1	2C	89	BC	DC	17	D1	47	99	FB	86	19	D9	FE	C6	E4	60	ED	DF	F6	91	11	11
114B901C00	1A	FD	C1	97	3C	43	F5	6C	FE	E9	78	F9	EA	1B	21	18	89	34	D4	BD	68	1F	5B
114B902000	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
114B902000	CO	AC	51	AB	7F	E5	05	4B	BA	93	8C	C7	45	B0	66	98	8F	00	10	EB	29	CE	00
114B902200	5D	BD	1D	DE	B0	87	67	90	DD	6F	4E	30	12	5D	DA	D6	0F	8F	26	85	04	58	FE
114B902400	33	7C	2B	34	C2	E6	4F	DF	FE	30	24	CO	76	D0	01	7E	EF	47	32	B6	DC	E1	0F
114B902600	81	29	D0	01	BA	80	17	84	23	DD	0D	04	03	3F	E0	DB	3D	57	33	87	7C	DA	69
114B902800	FC	96	FC	E9	E1	B0	DD	12	84	3C	6D	40	34	F0	8C	20	4F	10	FA	B7	9F	F6	76
114B902A00	4F	09	E0	B0	DD	1A	B1	DD	C3	1F	7C	40	70	20	92	1D	BD	17	A0	96	36	45	9E
114B902C00	E3	A5	F9	33	2A	56	0F	AB	88	35	C4	EA	63	45	FF	F2	70	47	58	07	20	95	0C
114B902E00	6E	BB	4B	D0	9E	39	FC	63	0F	F3	01	D3	87	79	8F	19	C4	8C	63	5E	01	7B	30

Sector 9,067,648 of 103,729,144 | Offset: 114B90003 | = 51 | Block: 114B90000 - 114B90003 | Size: 4

**Figure 30:** The header of the `smallsmallsmall` file downloaded from Canvas.