To know more about the profile and its function, I used it on the lists machine project. Based on the analysis of the profile, the Huffman decode symbol function took the most time. So did image color classes and functions. This shows that loading images and storing images takes the most time. This makes sense since images have lots of data to load and store for each pixel, and one image has lots of pixels. Besides, vectors and their operations took a lot of time. This is weird since we implemented linked lists for this project instead of vectors.

As for the functions we implemented, merge, destructor, sort, and split took relatively more time compared with other functions, especially merge and destructor. Therefore, we can consider improving the destructor and merge function to improve the time efficiency of the project. As for other functions we implement, they perform very well and do not take lots of time.

As for other functions and classes, find, get, and iterators took some time. This makes sense since we need these functions to get and modify the data of the images. As for the rest of the stuff, debugging and other codes that make sure the project runs correctly took large parts. However, these codes and functions did not take a lot of time. This makes sense since the compiler needs to call different functions to make sure the code can run correctly and they are fast.

After reading the timetable of functions, I started to look at the call tree of the program. The test cases took the most time since the test cases are parents and the root of the program tree. It is reasonable to take more time. Besides, it was clear that the image loading, storing, and printing took a lot of time. As for other functions, the time they took was similar to the previous paragraph. However, one good thing about the call tree of the program is that I can know the relationship between different functions and which children take more time than the others. With the help of the tree, I can know which functions are time-inefficient functions and their children, which can be helpful to improve the time efficiency of the function.

In conclusion, the profile is a great tool to show the time consumption of functions and classes and the relationship between them. It is good to show which functions are time-inefficient and which functions should be improved.