```
import argparse
import concurrent.futures
import hashlib
import os
import sys


def calculate_md5(filename: str) -> str:
    with open(filename, "rb") as f:
        return hashlib.md5(f.read(), usedforsecurity=False).hexdigest()


def check_single_file(expected_hash: str, filename: str) -> None:
    if filename.startswith('*'):
        filename = filename[1:]

    if not os.path.isfile(filename):
        print(f"{filename}: MISSING")
        return

    try:
        if expected_hash != calculate_md5(filename):
            print(f"{filename}: FAILED")

    except IOError as e:
        print(f"{filename}: {e}")


def verify_checksum_file(checksum_filename: str) -> None:
    if not os.path.isfile(checksum_filename):
        print(f"ERROR: The checksum file '{checksum_filename}' was not found.", file=sys.stderr)
        sys.exit(1)

    with open(checksum_filename, 'r') as file, concurrent.futures.ProcessPoolExecutor() as
executor:
        for i, line in enumerate(file, 1):
            line = line.strip()
            if not line:
                continue

            parts = line.split(None, 1)
            if len(parts) != 2:
                print(f"WARNING: Line {i} has an invalid format and will be skipped: '{line}'",
file=sys.stderr)
                continue

            executor.submit(check_single_file, parts[0], parts[1])


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument("checksum_file", help="The file containing the MD5 hashes and
filenames.")
    args = parser.parse_args()
```

```
    verify_checksum_file(args.checksum_file)


if __name__ == "__main__":
    main()
```