memcheck.h | Coursera

coursera.org/learn/interacting-system-managing-memory/supplement/RwCoz/memcheck-h

memcheck.h

memcheck.h

Sometimes we may want to interact with Valgrind's tools directly in our program. For example, we might want to explicitly check if a value is initialized at a certain point in the program (*e.g.*, as part of debugging an error about uninitialized values). Valgrind provides header files, such as**memcheck.h**, which contains a variety of macros for exactly this purpose. For example, we could change the function we were using earlier as an example of uninitialized values to

```
1
2
3
4
5
6
void f(int x) {
  int y;
  int z = x + y;
  VALGRIND_CHECK_MEM_IS_DEFINED(&z,sizeof(z));
  printf("%d\n", z);
}
```

Now, when we run this program in valgrind, we get the error message more immediately:

```
1
2
```

```
3

4

5

6

==12425== Uninitialised byte(s) found during client check request

==12425==    at 0x4007C9: f (uninit4.c:8)

==12425==    by 0x400811: main (uninit4.c:17)

==12425==  Address 0xfff000410 is on thread 1's stack

==12425==  Uninitialised value was created by a stack allocation

==12425==    at 0x400765: f (uninit4.c:5)
```

Many of Memcheck's features are available through these macros. Most other tools have similar header files for programs to interact directly with them. See http://valgrind.org/docs/manual/mc-manual.html#mc-manual.clientreqs for more details.