BUGS stands for:
Behavioural
Unreliability
Germination
Syndrome

1: Why are you being sent this message?

This message is being sent to every software company in the country. It is about BUGS. And everyone now needs to know the facts. It explains what BUGS are. How they spread. How serious a threat they are. And how they can be avoided. Because it has to deal with matters of software and hardware, you may find some of the information disturbing. But please make sure that everyone who may need this advice reads this message. The more people know about BUGS, the less likely they are to spread.

So, if you employ programmers, think carefully what they need to know. Whether you approve or not, many programmers do write programs, and some may experiment with assembler.

Even if you think your programmers don't, they will need advice because they may have friends who encourage them to.

2: Why should YOU be concerned about BUGS?

Any computer user can suffer from BUGS, depending on their behaviour. It is not just a hacker's problem. There is no cure. And they cause crashes. By the time you read this, probably 3000 computers will have crashed in this country. It is believed that a further 300,000 carry BUGS. This number is rising, and will continue to rise unless we all take precautions.

3: What are BUGS?

BUGS are caused by logical errors. These can attack a program's validation routines which normally help fight off bad input and corruption. And if this happens programs can then develop BUGS. They become unreliable and crash from errors they cannot detect.

4: How do programs become affected?

Because errors can be present in specifications and designs, this means that for most programmers the only real danger comes through using a faulty design. This applies to both top-down and bottom-up design. (It could also be that other forms of design are risky, particularly if specifications are taken at face value).

So the errors can be passed from design to design, design to program, and program to design. For those who use assembler, there is an added risk from sharing software interrupts or device drivers with an affected program.

Finally, sub-tasks spawned from programs which are affected have a high chance of inheriting the errors.

5: How can you protect yourself from BUGS?

Most programs which have errors don't even show it. They may look and behave completely normally. So you cannot know which ones are affected and which are not. To protect yourself, follow these guidelines. The more programs you write, particularly assembler programs, the more chance you have of encountering one which is affected. It is safest to stick to one program.

FEWER PROGRAMS, LESS RISK.

Unless you are sure of a program, always use a debugger. This will reduce the risk of getting errors.

USE DEBUGGERS FOR SAFER PROGRAMMING.

It's also best to use a source-level debugger. Low-level debuggers can reduce the protection. Ask your team leader for advice.

Anyone who uses assembler programs should not use software interrupts. If you ever do, never share interrupts (or device drivers, IO routines, etc.). You could be introducing errors directly into your program. It is extremely dangerous.

DON'T USE SOFTWARE INTERRUPTS, NEVER SHARE.

6: If you think your program has been affected.

If you think your program may be affected, go to your QA person for advice about having a test. Or go direct to your team leader for confidential advice and a test if you wish. If your program does have errors, they'll let you know and give you help and support.

7: What about resident software?

It is NOT safe to use Hot-Key programs, comms packages, or on-line dictionaries unless you know they are bug-free and have been tested. Nor is it safe to share a routine or include file with a program which has been affected. These things could give you errors through affected routines.

8: What can’t you get BUGS from?

The government’s clear advice from the software industry is that you cannot get BUGS from normal use of an affected program. You cannot get them by reading listings. Nor is there any record of a program having become affected through sharing comments. There is no danger in sharing floppy disks or backups. Nor can a program be affected by using network drivers or printers.

When integrating software, standard precautions protect programmers, integrators and users.

Exporting routines is safe. All the declarations are only used once. All the routines used for library builds are rigorously checked.

9: How safe is it in other companies?

BUGS exist throughout the computer industry. In certain companies a large number of programs have them. So it is even more important to follow the advice in this message if you are visiting another company. Otherwise, if you do some work on an unfamiliar program, you might inadvertently affect your regular work when you return.

Again, in some companies public routines are not checked for BUGS. In those places where errors are widespread do not, if you can possibly avoid it, use routines from a contract programmer.

Also, in certain start-up companies, routines may not be properly debugged. If you can, avoid anything using interrupt bounces or patches.

If you have any worries about this discuss them with your team leader.

10: Do you need more information?

The true picture about BUGS is that, at the moment, relatively few programs are affected. Those most at risk are assembler programs which have common code with other assembler programs, programs which share software interrupts, and users of these programs.

But errors **ARE** spreading; and as they do, so the risk of sharing a routine with a program which is affected increases.

Ultimately, defence against BUGS depends on all of us taking responsibility for our actions.

BUGS

DON’T LIVE IN IGNORANCE

GOVERNMENT MISINFORMATION 1987