Notes:

If you have worked through this manual you will have a good understanding of the basics of computer science and you will be able to write a computer program to solve simple problems. This bears repeating: you can write a set of instructions to make a computer do something. This simple statement embraces a set of powerful thinking skills that include logical reasoning, problem solving, algorithm design and much more. As of now, you are no longer just a consumer of digital products – you are a creator! This is something of which to be proud.

We hope that this brief introduction has shown you how useful, exciting and fun computing is as a hobby or career. If, at any time while following this guide, you grinned to yourself or you shouted downstairs, "Come and see what I've made!" then we have done our job – this is, after all, why the Raspberry Pi was made. If you don't go any further with your Raspberry Pi journey then you at least know what all the fuss is about and that it's not as hard as people think. It is quite probable, however, that you have caught the computing bug and would like to learn more.

So - what now? There really is no right or wrong way to continue your exploration of the Raspberry Pi. It all depends what interests you. It could be as simple as getting better at your chosen programming language, or improving your Linux command line skills. Small projects, such as setting up a home media server or writing a simple game, are also a good way to start. At some point, most people will want to have a go at interfacing the Raspberry Pi with the outside world (this is one of its strong points) using a breakout board such as the Gertboard, or by making their own interface. Again, a simple project such as monitoring temperatures or controlling a cheap robot is the place to start.

As your understanding of the Raspberry Pi grows and your programming skills improve, you will find that your projects get more and more complex, and that the Raspberry Pi becomes a serious tool for experimentation and creativity. Keep practising – one of the beautiful things about programming is that you can take a blank text file and create something that previously only existed in your head. What you make is limited only by your imagination.

Where can I get help, ideas and inspiration?

The best resource to help you continue your computing journey is the web. It's vast, full of people with large brains, chock full of diverse and arcane knowledge, and unlike this manual it gets updated frequently. To save time we were just going to steal a huge list of links from somewhere, paste them here and make ourselves scarce. But our editors told us off. Here then are our links with a few comments[1]. It's only a start, but should save you some time trawling the web.

[1] *Disclaimer: if a resource is mentioned here, it does not mean it is the best in class. Likewise, if a resource isn't mentioned, it doesn't mean that it's no good. We have no agenda apart from getting people into computing and we have no affiliation with any of the sites or people mentioned. We do rather like the Raspberry Pi Foundation, though.*

It concentrates on programming because the Raspberry Pi was made to encourage a new generation of programmers. While there is much more to computing than programming, it is a very good place to start.

Notes:

General resources and help

The first place to look is the official Raspberry Pi website and forum (***http://raspberrypi.org***). There is a lot going on in the forums and there are subforums for specific topics such as GPIO, programming, operating systems and education. The members are a friendly bunch who collectively have a huge amount of experience and expertise. They would be happy to answer any questions, help you with problems and point you in the right direction. If you have a question about your Raspberry Pi or are trying to get something to work then this is probably the best place to start.

The Raspberry Pi wiki at eLinux (***http://elinux.org/R-Pi_Hub***) is also a great source of information and is constantly updated. It's a wiki, so feel free to add to it and improve it as your knowledge grows and your expertise widens!

Perhaps the most useful thing that you can do is to join some forums (no, not *fora*) and talk to people. Get involved, ask questions, help others. Above all keep on computing and have fun!

Programming

The first question that those new to programming ask is, "What is the best language to learn?" Put simply – there isn't one. Just discussing programming languages would fill another volume (and would make you claw frantically at your keyboard in bored rage, like some deranged man-mantis).

Each language has its advantages, disadvantages and particular uses, and all programmers have their favourites. It can all get a bit, "My dad's better than your dad", which isn't that helpful if you don't know where to start. Ultimately, it's the underlying computational concepts that you learn while programming that actually matter. So our advice at this stage is this:

The first language you learn is unimportant. Pick one and get programming!

[Update: the author of this statement has since had to go into hiding, after he was attacked in his local supermarket by a horde of angry programmers, who hurled turnips at him and called him an "ill-educated homunculus".]

So, pick a language and get started. Unless you have reason not to, then continuing with Python seems sensible. Eventually, you'll learn other languages but by then you will know what you want to learn and why.

Raspberry Pi-specific resources

Liam Fraser has made a series of Raspberry Pi-specific YouTube tutorials, which you can find at ***http://goo.gl/MM9hA***

The Magpi, the free Raspberry Pi Users' magazine (***http://themagpi.com***) has some excellent beginners' articles and tutorials.

Python online tutorials, resources and references

The official Python site keeps a huge list of quality resources both for beginners (***http://goo.gl/MMo5G***). It should keep you busy for some time.

Online courses

Free online courses by top universities have taken off recently, and they teach many computing-related topics, such as programming, electronics, artificial intelligence and logic. The courses are pitched at college/university level but the beginner courses could be tackled by younger students.

The courses are typically delivered via video lectures and handouts, with accompanying exercises. Many have definite start and end dates, with homework deadlines and formal exams, although some – such as the Udacity CS101 course – are now 'open' and you can start (and finish) any time. These courses are ideal for people who want to do some more in-depth stuff and would like guided tuition with a specific target.

Currently, the main providers are:

Udacity (*http://udacity.com*)
The *Introduction to Computer Science* (CS101) course needs no prior computing or programming experience. It takes you from nothing to building your own search engine (though we don't think that Google should be too worried!).

Coursera (*http://coursera.org*)
Coursera offers 120 courses across 16 categories. *Computer Science 101* is the obvious beginner's computing course but hasn't started as of October 2012. One advantage is that it has a self-study option where you proceed at your own pace (but you don't get a certificate).

edX (*http://edx.org*)
A collaboration of MIT, Harvard and Berkeley, edX currently offers two introductory computing courses: *CS50x Introduction to Computer Science I* (Harvard) and *6.00x Introduction to Computer Science and Programming* (MIT) due to start in October 2012.

Notes:

Online practice and tutorials

Notes:

More informal ways of learning to program online include Codingbat (***http://codingbat.com***) and Codecademy (***http://codecademy.com***). Codingbat is a series of programming challenges that help build strengths in specific areas, such as string handling and logic. Teachers can track student progress, and tools include a useful graph that logs incorrect and partially correct attempts at a solution (this makes it hard to cheat!). It has exercises for Java and Python. Codecademy is a great place for anyone to start programming from scratch. It's user friendly and it tracks your progress. It teaches Javascript but a Python "module" has recently been added.

Less formal online challenges can be a big motivator. These exist for both specific languages, such as the addictive Python Challenge (***http://www.pythonchallenge.com***), and for specific problem-solving, such as the maths-based Project Euler (***http://projecteuler.net***). A full list can be found on the Raspberry Pi forums at ***http://goo.gl/n7ej4***.

Beyond online learning

Online practice and exercises are useful but the best motivation to learn to program is to actually make something. This means starting a project with a specific purpose. Another way to improve is to work on projects with other people. See if there is a local Raspberry Pi or Linux user group near you. If you are at school or college join the computer club (or set one up).

Look at other programmers' code and try to understand it. Change it for your own purpose or try and improve it. If it's broken, try and work out why. Don't forget that programming at this level should be fun – extending yourself is one thing but don't get so frustrated that you give up. Talk to people on forums and have a look at how they have solved the problem, there's always more than one way to tackle it.

The end

We hope that you've found this manual useful and that learning to program has been as mind-expanding as it was for the authors when they wrote their first lines of code. Computers are amazing things and being able to control them is an amazing ability.

Keep on programming, hacking, playing, experimenting and creating. And above all, have fun!