How and Why to Write a Program

While some people are more than content to browse through the enormous list of tested and proven programs available these days, or (often needlessly) pay someone to write programming for them, many feel the need to reinvent the wheel and write their own programming. Well I am going to be the asshole to tell you that you should probably think twice about that. Call it gatekeeping, call it condescension, but the truth is most people do not have the experience needed to write a personalized program that will work better than a good generic program. And even more people do not really need one. This is not me telling you that you have not earned the right, you can absolutely do whatever you want, this is me telling you that your results will likely suffer from trying to do your own programming instead of just running something tested and proven.

This warning comes from a place of personal experience. My first outing in writing my own programming resulted in the least productive year of training I have ever had. I ran something I wrote for over a year and I ended up with jack shit to show for it besides a token 10lb chip on my squat PR. I was over three years into training, and had just finished two plus years of nSuns LP and CAP3. I felt that I had outpaced both programs, but did not want to shift to something totally different. So I took a very loose understanding of Wendler's 531 BBB (Boring But Big), and hit it with a hammer until it was a pile of scrap that covered 6 days a week. I did not know what I was doing, why I was doing it, and I had only ever experienced one flavor of programming. Turns out that 531 BBB is not meant to be a 6 day program.

Doing 531 sets, plus 5x10, plus accessories 6 times a week, doubling up on SBD and cutting out OHP is not a good programming decision. But I was not going to let that stop me, I was going to run that ship fully aground over the course of a year and wonder why I was not getting anywhere. I cannot stress enough that I did not know what I was doing and that I had zero business writing a program after only three years of training in only one style. Do Not Be Like Me.

I lead with this anecdote so that you can understand that this write up is not me telling you that I am special and can write programs and you are not so you can't. I am relaying my own personal failings in that endeavor so that you might understand why it might not be a good idea for you.

So what has changed between then and now? I have an additional 5+ years of experience with a variety of programs, including my failed program. I have a much firmer grasp of what does and does not work for me, and what my training philosophies are. I have much more nuanced ideas of what my goals are and how I can reach them. I am thinking in programming blocks, not just in perpetual, consistent training. I am much more experienced and have a much greater knowledge of how things work, and how they don't work. The only thing that can give you this is time under, over, or around the bar. I do not believe you can effectively 'study' these things from a purely, or even predominantly theoretical perspective, particularly when it comes to you as an individual.

So what is the best way to accumulate this experience? Train with a large variety of someone else's programs. Try to pick programs that are radically different, or at very least from different people in different disciplines. Even if you want to focus on programs that generally gravitate around SBD, or some other discipline, you should at least run a lot of different SBD based programs in order to broaden your perspectives on how to train these movements. No two experienced coaches are going to have the same outlook, and by experiencing their programming you gain the benefit of their unique approach to training. As you gather this firsthand experience, pay attention to what you like and don't like. What works for you and what does not. These are the aspects you are going to want to include, or exclude, in your future programming.

So now you have run at least 10 different programs, you've been at this for quite a few years and have a pretty solid grasp on how you operate, and you have results to show for it. Are you ready to write a program? Maybe. Should you? Well that is a totally different answer. Ask yourself these questions, I include the answers I had when I was writing Rip And Tear (you can read more about it in the write up of the same name):

"Why are you writing a program":

The answer probably should not be 'Because I just want to'. I can't stop you if this is your only reason, but it's not really a good one. There are people better at this than you, and they are offering up their programs. You will almost certainly be better off just picking one that aligns with your needs and using that.

A better answer to this question is 'Because nothing I have found quite meets my current needs'. Now you are getting somewhere. Generic programs are just that, generic. This does not mean that they are bad, but it means that they are meeting relatively broad needs so as to apply to a wide audience. A custom program does not need to do this, it can focus on as narrow and obscure a goal as you want it to. It can be tailored to your needs and wants. Why did I write Rip and Tear? Because I wanted to run PH3 again, but PH3 was not really compatible with me. No other program that I knew of had the parts of PH3 that I wanted but with more compatibility.

"Can you just modify an existing program":

As a follow-up, if you have a unique need, can you fulfill it by just taking a program and making some minor changes? As far as I am concerned there is no shame in working off someone's programming homework and changing it just enough that the teacher does not notice. These modifications can be designed to shift the program's focus to favor your goals, work around injuries or other physical limitations, work around equipment limitations, or include things you enjoy that the program as written does not include.

I tried this with PH3 before eventually writing Rip And Tear from the ground up. Minor modifications still didn't quite get it where I wanted it, and I was approaching the point that I had so many changes I wanted to make it was easier to just start from the foundation.

”What is your program going to do, and how”:

Even if you have identified that no existing program can meet your needs, with or without modification, you still need to be able to explain to yourself how your proposed program will meet those needs. How are you going to structure your program so that it helps you reach your goals? What is it going to look like, in broad strokes, and why are those strokes going to lead towards your success? If you can’t explain why your program will work, it probably isn’t going to do so.

In my case this was an easy question to answer. I wanted to lift heavy SBD, and I wanted to do it a lot. So all my program had to do was contain a lot of heavy SBD sets, and all other decisions just had to support and facilitate that simple goal. If your goal is more specific or based on an actual result, you might have a harder time with this question.

”What are you basing your program on?”:

I don’t think that anyone should just sit down with a blank piece of paper and completely create a brand new program from scratch. The experience you are leveraging to write this program came from other programs. Use them. Take chunks of other programs shamelessly. Nothing is original, even if you make up everything in your program from scratch it’s still going to end up copying parts of someone’s programming. Don’t make more work for yourself, work with the enormous body of existing programming.

When writing Rip And Tear I worked from the foundation and core essence of PH3, and included aspects from other programs that I enjoyed, such as the raising set/lowering rep progression that I took from Greg Nuckol’s 28Free Program Builder.

”What are you going to call your program”:

All the good programs have cool names based on cool shit like superheros, heavy metal, giant robots, or other intense shit. If you are going to write a program you have to name it something suitably intense to properly represent how powerful it is. I named my program Rip And Tear because it’s intense as shit.

If you have established that you can make a program, should make a program, and know what’s going into it, here are a few final thoughts:

Everything does not need to have precise reasoning. When writing Rip And Tear I just picked percentages because they looked reasonable. I picked 2.5% jumps because it seemed like a good number. Why did I up the jump to 5% for the 3rd week? I accidently typed it that way in

week three and decided that it looked alright to me. This was all totally fine because the precise percentages don't really matter that much. A little lower and I would probably just get higher AMRAPs and have more juice for accessories. A little higher and it would be the opposite. Neither is a big deal.

Along similar lines you should have a good sense of what your program is going to do, and a reasonable sense of how it's going to do that, but the precise mechanisms aren't super relevant. You don't need to stress over having a double undulating bilateral progression scheme or a reverse pyramid step back looping progression, or whatever the influencers pretending to be eggheads are talking about. Is the program moving generally forward in some manner? That's probably good enough for progression. You should know several progression schemes that you like from the experiences you gathered. Just pick one.

Adding specific movements to your program? Just pick some, I promise the choice between a banded floor press versus a chained block press won't make or break your programming. Focus on the large scale, not the small scale. Your foundation matters, what color you paint the walls does not.

Don't be afraid to audible. If you realize a third of the way in that something is not working change it. You aren't going 'off program', you wrote the damn thing, you are *editing* it. If you have the experience necessary to write a program you have the experience necessary to make judgment calls to change things mid-program. Frankly the latter requires less experience than the former.