言語処理系の開発にあたって

954480 岡本悠里

第1版　2022/01/30

この課題を行って会得したことや感想などを三千字(3,000字)以上

この授業で学んだプログラミング言語は、パラダイムを学ぶものとなった。プログラミング言語には関数、論理などスタイルが決まっている場合が多いと感じた。マルチパラダイム言語でもどのパラダイムを採用するかを決めている。今回の課題で言語を作るとなった時に、新しいパラダイムを考案しなければ新しく言語を作る意味がないと思い、新しいパラダイムを考え出そうとした。言うまでもなく、苦戦し、パラダイムを作り出すとは何なのかについて考えざるを得なくなった。いくつかのパラダイムを眺めているうちに、パラダイムというのは作ろうとして作れるものではないのではないかと考えた。困ることや要望があってそれを解決する手段として考え出されるという流れを踏み、先にパラダイムを作り終えてその後にどういった要望が叶えられるかという順ではないのはないかと思った。つまり、この課題を取り組むには日頃困っていることなどを洗い出し、それらを解決するような言語を作るという流れが自然ではないかと考えた。よって、今回はパラダイムを考え出すことまでは出来なかったが、英数字の羅列が怖いと思う困りごとを想定し、これを解決するような言語を作ろうということになった。更に、自分の経験から値の代入または束縛において1回生の時に感じた「=」を使う違和感について解消しようと考えた。統合できないものに=を使う事に違和感を感じる上に、代入と束縛の違いが全く理解できなかった。視覚的に最初から荷札だと思わせてしまえば良いでのはないかと考えた。つまり、この言語を使うユーザを玄人ではなく素人とした。yaccやlexを用いながら自分がデザインした言語を実装していく体験ができるということはもちろんなのだが、実際に世の中に出た際に誰が使うのだろうかということを考える機会になった。先生の授業を聞いていきなり全部実装するというのはやめて、少しの機能から実装しなさいというアドバイスをもとに最初は定義と足し算引き算しかないものを作った。たかが足し算引き算だが、実装は困難を極めた。この僅かな機能であっても起こりうる想定外の事態がたくさん考えられた。決めた文法に沿って定義が行われるように、その定義を踏まえて正しく計算できるようにするだけでもこの場合はうまくいくべきであり、この場合はエラーにしなければなど不具合の削減が難しい。今まで行ってきたソフトウェア開発でのテストとは違ったテストであると感じた。単体テストや結合テストなどだけでなく、そもそもの言語を作っているので、起こりうる入力パターンを確実に掬い上げる必要がある。テストケースをできるだけ多く作り出し、あらゆる脅威に対応することの重要性と難しさを学んだ。私は最初、lexで文字列の定義をクオート( ‘ )で囲まれたすべての文字を文字列と定義していた。考えられる入力を考えているうちに、クオートが含まれた文字列が渡された時にうまくいかないことに気づいた。文字列でクオートを使いたい時に困ってしまう。たかが定義、たかが文字列、たかがprintだがこのように不具合が発生する。結局、\’と入力しエスケープさせることでクオートも文字列として定義することができるようになった。少ない機能にもかかわらず次々不具合がでてきた。一気にwhileやらforやらifを注ぎ込んで作成していたら処理系を作ることにも至らず、一度も動くことがなかったのではないかと思う。時間や技術の限界で足し算引き算printほどの小さな機能しか織り込めなかったが、プロンプタをターミナルに出し、足し算や引き算を行うことができる言語を作ることができた。多くの言語がバージョンアップを重ねながら機能を増やし、不具合を修正していくように、この言語でも完璧に1から作ろうとせずに、少しずつ段階を踏みながら成長させていくという考えが重要であると考える。

　言語を作成するにあたって、今回もドキュメントを作ることにした。ソフトウェア工学からの積み重ねによってドキュメントを作る流れがどんどんスムーズになってきているように感じる。初めて作成した時はかなりの時間を費やし、作ることに抵抗もあった。苦しい時を過ごし、未熟ながらも1度完成させ、更に長期課題1でもドキュメントを作成するなどといった反復により、ドキュメントを作る流れが染み付いてきた。誰がどのように作ってきたのかをきちんと示し、どのような設計図を経て完成まで至ったのかを示すことで、万が一の時に備えることができる。たとえ課題が完成しなくても出来た部分までを示すことができる。誰かから権利を侵害されるような行為を取られたとしてもどちらが先に作ったもので、オリジナルがどちらかを示すことができる。ソースコードをただ提出するだけではなく、開発の取り組み方や変遷を示し、どこまでテストしたのかを示すことは自分達が行ったことを他者に伝えるだけではなく、今後の自分にも役立つのではないかと気づいた。今回ドキュメントを作成するにあたって過去のものを参照した。過去の自分や過去のチームメンバーが何をしていたのかを振り返ることができた。チームメンバーはプロジェクトごとに変動する。今回は人数もかなり減ったため、自分の担当する部分も増えた。過去のドキュメントを見ると、今まで自分がやっていなかった仕事を誰がどのようにやっていたのかを知ることができる。これは多いに助けになった。未来の自分や他者の助けになる可能性を持つものがドキュメントであるのではないかと思う。今回の課題を含め、今まで自分とその仲間が作ってきたドキュメントが誰かの助けになるのかもしれない。誰がどれだけやったのか、どのように取り組んだのかという「課題」という狭い枠組みの中での「評定」を公正につけてもらうというその場だけ有効なものではない。記録し、誰かに伝え、遺すという行為の重要性とそのやり方について体感できた。このドキュメントを作るという行為は他の教科に活き始めている。全く他の教科でも、グループワークやプロジェクトのような長い時間をかけて取り組むようなものでは、なるべく記録を残し、作業過程を”レポート”するようになった。結果主義ではなく、経過まで報告するのがきちんとした報告であるという考えに変わりつつある。そしてその残し方は一連のドキュメント制作過程で身についてきている。リポジトリでバージョン管理を行い、いつどの機能が完成したのかという記録をつけることで、ドキュメントもより書きやすくなる。様々な手段を使いながら、記録に残していくという武器を会得しつつあると感じた。