Biostrings ISC Proposal 2024

Aidan Lakshman

2024-03-25

Project Team

Aidan Lakshman - PhD Candidate, University of Pittsburgh. More information is available below in the "People" section.

Contributors

Erik S. Wright - Associate Professor, University of Pittsburgh (Letter of Support)

Hervé Pagès - Bioconductor Core Team, Biostrings Maintainer

Signatories

From the ISC: "This section provides the ISC with a view of the support received from the community for a proposal. Acceptance isn't predicated on popularity but community acceptance is important. Willingness to accept outside input is also a good marker for project delivery."

If you are supportive of this project and interested in signing on, feel free to email me (AHL27@pitt.edu) or open an issue on GitHub.

The Problem

Biostrings is an essential package in the R ecosystem, with over 1 million installations from Bioconductor per year (Fig. 1). A total of 226 Bioconductor packages depend on Biostrings, 254 import Biostrings, and 43 suggest Biostrings. Despite this success, Biostrings has been unsupported financially for over a decade. Package maintenance currently relies upon a small patchwork of volunteer maintainers who struggle to keep pace with basic maintenance. This has resulted in the accumulation of technical debt, longstanding bugs, and insufficient support to implement planned enhancements. While many of these issues have been outlined within the Biostrings package, there have been few developers willing to learn the internal code structure of Biostrings to be able to take over maintenance.

As an active Biostrings user and contributor, I have discussed this issue extensively with the package maintainer, Hervé Pagès (see attached Letter of Support). In collaboration, we have drawn a roadmap to sustainable long-term maintenance of the Biostrings package. Here, we present a path forward and request funding support for the labor required to implement it. In this vision, I will take over primary maintenance of the Biostrings package and implement a set of fixes and extensions that put Biostrings on a path toward sustained success.

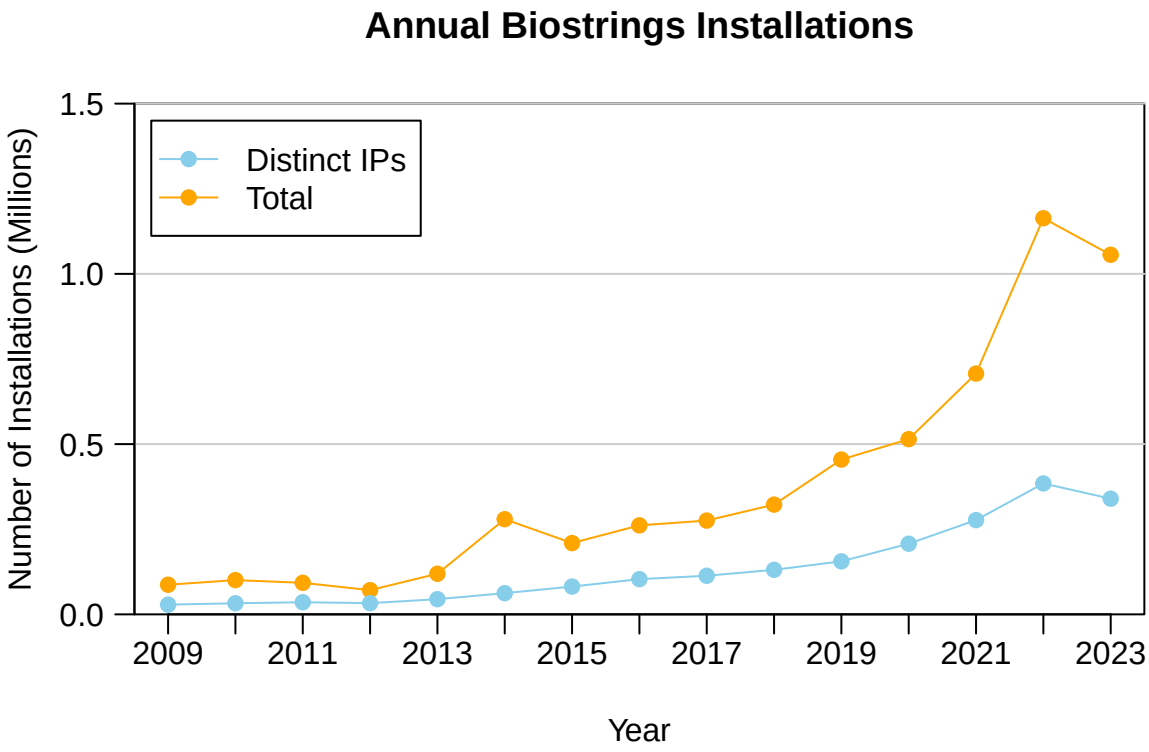

Figure 1: Figure 1. Yearly Biostrings Installations by total and unique IP address.

The Proposal

Overview

Biostrings is a core Bioconductor package providing efficient containers for storing, manipulating, and analyzing biological sequences. Biostrings is **the** method to access biological sequence data in R–nearly every analysis working with genomic data depends on the Biostrings package to handle sequencing data. Presently, Biostrings maintenance is hindered by (i) lack of robust testing and numerous open bug reports, (ii) incomplete implementations of critical functionality, and (iii) input/output functionality that is becoming outdated with newer technology.

This project proposes to clear out this accumulated technical debt by addressing open issues, implementing robust tests for long-term sustainability, improving user experience, and adding features that will keep Biostrings relevant for modern sequencing technologies. For end-users, this will result in numerous bugfixes, a host of new features to support genomic analyses, and a variety of performance improvements to bolster R as one of the top programming languages for bioinformatics. For developers, this will make the Biostrings package more sustainable, allowing for more community contribution and faster bug resolution in the future.

In summary, this proposal details a new era for Biostrings. The project will transition maintenance to a new developer, and in the process, ensure the package is robust and maintainable for years to come.

Detail

There are three categories of changes needed to sustain the utility of Biostrings in the long-term. First, Biostrings needs a better testing infrastructure to support future improvements. Implementing this testing framework will go hand-in-hand with addressing existing bug reports, most of which are straightforward but require effort to complete. I anticipate up to 4 months for this first Aim. Second, the Biostrings package contains a list of to-do items that have gone unaddressed for years due to insufficient developer time. Prior to this proposal's submission, I worked with Hervé Pagès to remove outdated items and narrow this list to elements that are the most important for Biostrings' long-term success. These tasks all deal with incomplete or confusing implementations of string matching functions, and I anticipate that they can be resolved within 6 months. Third, modern DNA sequencing technologies have advanced markedly since Biostrings was first introduced. Changes to import and export of sequences are needed to sustain the project long-term. I expect these changes will take 2 months. These three categories of changes are described in more detail in the three Aims below.

Aim 1: Resolving bugs and increasing robustness The goal of this aim is to prepare the Biostrings codebase for future improvements and facilitate ongoing maintenance. Biostrings has had minimal maintenance for the past 5 years, and as a result lacks many processes that ease continued development. The goals of the Aim are the following:

1. Clean up outdated documentation and bug reports. Many internal documentation files and scripts have `TODO` tags or warnings that are out of date. Additionally, many bug reports on GitHub have been resolved but remain open. This makes ongoing maintenance challenging, as it requires additional developer effort to determine if a bug still exists before addressing it. This task will clean up the codebase, update outdated documentation, and clean up resolved bug reports.
2. Implement unit testing. Much of this project depends on improving and adding additional functionality to Biostrings. However, as mentioned in the Problem Definition section, Biostrings is a critical package with over a million downloads per year. As such, robust unit testing is essential to ensure that other proposed changes do not break existing functionality for end users. Robust testing suites would also make it easier for maintainers to review community-contributed code, thus making it easier to involve the community in Biostrings.
3. Resolving outstanding bug reports and old TODO items. While many bug reports on GitHub are resolved, there are many that remain unaddressed. This task will resolve any remaining bug reports. Part of this task will involve reaching out to bug reporters for additional detail, as some bug reports are years old without any updates (see here or here examples of very old bug reports). This task will

also address some small high-priority tasks on the Biostrings TODO list, since they overlap with many of the GitHub issues.

The Biostrings repository contains a TODO list with many tasks, ranging from small bugs to long-term feature requests. Task 3 will address the small tasks from this list that overlap with existing GitHub issues, specifically the stack overflow issues relating to `XStringSet` comparisons (mentioned in the aforementioned Issue 51), ensure `validObject()` correctly works on `XStringSet` objects, and implementing missing `patternFrequency` methods for `XStringSet` and `XStringSetViews` objects.

Successful completion of this Aim will result in a cleaner Biostrings GitHub repository, resolution of outstanding user-submitted bugs, and a robust testing pipeline for future submissions. I expect this Aim to take three months; most of the GitHub issues are relatively quick to fix, so the majority of the time will be dedicated to building a robust testing infrastructure. The outstanding TODO items are also similarly quick to address–longer issues will be reserved for Aim 3.

Aim 1 is scheduled to be addressed first because it includes checks for later work and fixes for user-identified issues. I see these improvements as the highest priority for the Biostrings codebase as a whole.

Aim 2: Fixing String Matching The remainder of items on the Biostrings TODO list fall into three categories: "nice to have" feature requests, low-priority issues, and problems with string matching. The first two of these categories can be delayed to future work, but the third category directly impacts users. This aim will focus on cleaning up the string matching libraries of Biostrings.

Matching strings is a common task in bioinformatics, and, while Biostrings does provide a host of tools for string matching, many of Biostrings' tools are underdeveloped and/or confusing to use. For example, the following interaction is clearly not ideal:

```
library(Biostrings, quietly = TRUE)
```

```
## Error in library(Biostrings, quietly = TRUE): there is no package called 'Biostrings'
```

```
pd <- PDict(mkAllStrings(DNA_BASES, 3))
```

```
## Error in PDict(mkAllStrings(DNA_BASES, 3)): could not find function "PDict"
```

```
strs <- DNAStringSet(c("ATGCATGCA", "ATGATCATGA"))
```

```
## Error in DNAStringSet(c("ATGCATGCA", "ATGATCATGA")): could not find function "DNAStringSet"
```

```
matchPDict(pd, strs)
```

```
## Error in matchPDict(pd, strs): could not find function "matchPDict"
```

```
vmatchPDict(pd, strs)
```

```
## Error in vmatchPDict(pd, strs): could not find function "vmatchPDict"
```

Additionally, string matching doesn't support the widely used IUPAC ambiguity codes:

```
library(Biostrings, quietly = TRUE)
```

```
## Error in library(Biostrings, quietly = TRUE): there is no package called 'Biostrings'
```

```
patterns <- DNAStringSet(c("ATR", "GCY"))
```

```
## Error in DNAStringSet(c("ATR", "GCY")): could not find function "DNAStringSet"
```

```
pd <- PDict(patterns)
```

```
## Error in PDict(patterns): could not find function "PDict"
```

Searching for strings containing some ambiguities is a common task, and not supporting this functionality limits the usefulness of Biostrings for bioinformatics.

This Aim will work on improving the user experience for string matching functions by adding in capabilities that users expect to find and resolving cryptic error scenarios. More specifically, this will address the following tasks:

1. Allow for variable width matching in `matchPDict`. The current implementation relies strongly on the notion of a "Trusted Band". This implementation is very difficult to understand for end-users, leading to confusion on how to use it and seemingly cryptic error messages. Trusted Bands can become a purely internal argument, removing a significant amount of user burden. This would also allow users to more easily search for variable width patterns in data, which is a common task.
2. Allow for ambiguity codes in search strings. This has a variety of use-cases–a representative example is searching for a particular amino acid in a nucleotide coding sequence. Many amino acids are encoded by multiple codons, and the current implementation requires specifying all of these codons individually. For example, searching for Valine could be done by simply looking for `"GTN"` rather than `c("GTA", "GTC", "GTT", "GTG")`. This would also improve internal performance by decreasing the number of searches performed.
3. Resolve the circular error of `matchPDict` and `vmatchPDict` shown previously. This involves implementing `vmatchPDict` and removing these error messages.
4. Implement small helper functions for `PDict` objects to improve user experience. Specifically, implement `reverseComplement`, `duplicated`, and `patternFrequency` for `PDict` objects, and add the `skip.invalid.patterns` argument for `PDict` that has been promised in documentation for years. This will also add in helper functions to clarify available algorithms for `matchPDict`, which are currently relatively hard to understand.

Aim 2 is the longest aim, and is anticipated to take 6 months. While many of these tasks are nontrivial, much of the initial plans to implement them are detailed in the Biostrings TODO file. Many of these tasks also build off of existing functionality (e.g., `vmatchPDict` for Task 3 will primarily leverage the existing code in `matchPDict`). The remaining tasks in the TODO file ("nice to have" and low-priority items) will be addressed after the conclusion of this grant (see "Future Work" for more discussion on these tasks).

Aim 2 is positioned second due to it containing large tasks that will improve user performance, but that are less critical than open bug reports. The conclusion of Aims 1 and 2 will result in a significantly improved user experience for R's bioinformatics community.

Aim 3: Enhancing input/output for modern sequencing technologies Biostrings was initially developed during a time when sequencing produced megabases (~1M nucleotides) of data per run. However, modern sequencing technologies easily produce gigabases (> 1B nucleotides) per run. Hence, Biostrings' input/output needs improvement to scale alongside next generation sequencing technologies. At present, Biostrings can only read and write sequences from gzip compressed files. There is an `open_input_files` function that allows reading sequences in batches, but it does not use the standard R connections interface and is cumbersomely slow on large compressed files.

To enhance Biostrings, I will add functionality for reading from standard gzip, bzip2, and xz connections in R. This will involve overhauling the `readXStringSet` functions within Biostrings. Furthermore, I will enable writing to alternative output file compression types (bzip2 and xz), while allowing for different compression levels. While a `compression_level` argument in `writeXStringSet` does exist, it is unused by the function. I will focus on improving the speed of reading and writing from files so that large file sizes are no longer problematically slow. Collectively, these enhancements will propel Biostrings (and by extension, the R programming language itself) into the future of big biological data.

This Aim is placed last due it being oriented towards the future of Biostrings. Aim 1 adds robust testing, Aim 2 addresses existing problems with functionality, and Aim 3 prepares Biostrings for the future.

Project Plan

I plan to complete the proposed work in a one-year time period. As funding commences June 1, this means the duration of the proposal is June 1, 2024 - June 1, 2025. A summary of the timeline of aims is included below, and a detailed description follows.

Since the package is hosted on Bioconductor, large milestones are already clearly defined. Bioconductor releases new versions in October and April, which will each act as milestones for delivery of the first two Aims. This grant will conclude shortly before useR! and the annual Bioconductor conference (typically held in July), allowing me an opportunity to highlight the work done and the ISC program at the end of my award.

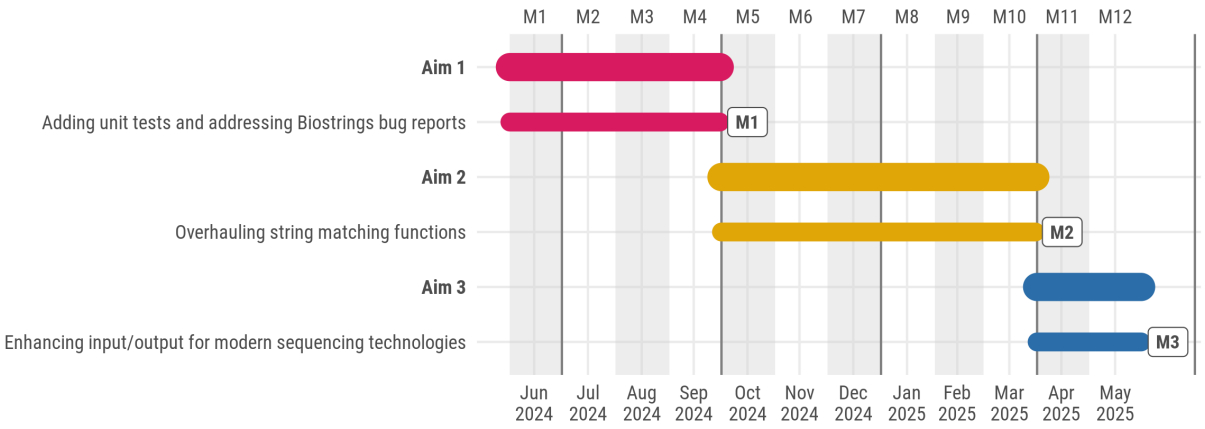

Figure 2: Figure 2. Timeline of aims and milestones (M) throughout the project. M1 and M2 are Bioconductor releases 3.20 and 3.21 (respectively), and M3 is the conclusion of the grant.

Start-Up Phase

This project has a short start-up phase. I have already coordinated with Hervé Pagès to identify critical tasks for each of the Aims (see "Proposal"). The codebase and dissemination method have already been created (GitHub and Bioconductor, respectively), and I have acquired write access to the codebase from Hervé Pagès. Additionally, I have experience contributing to Biostrings in the past, so I am relatively familiar with the codebase and contribution pipeline. As a result, I would be ready to begin work on Aim 1 from the moment the grant is awarded.

Technical Delivery

Aims will be completed on the following schedule:

Aim 1: Finished by October 2024, in time for Bioconductor release 3.20.

Aim 2: Finished by April 2025, in time for Bioconductor release 3.21.

Aim 3: Finished by June 2025 and merged into Bioconductor version 3.22 (development). Present on improvements at conferences, finalize changes in time for Bioconductor release 3.22 in October 2025.

Specific deliverables are as follows:

- June 1: Project milestones added to this project's GitHub project tracker
- June 30: Biostrings GitHub issue tracker cleaned up. All open issues will be verified unresolved bugs.
- Aug. 15: Testing framework for Biostrings implented, test suites included in `tests` folder in codebase.
- Sept. 31: Open bugs from Biostrings GitHub resolved. Aim 1 complete; Biostrings release for Bioconductor v3.20 finalized.

- Nov. 30: Variable width-matching for `PDict` objects completed, Trusted Bands phased out.
- Jan. 31: Support ambiguity codes for `PDict` objects.
- Feb. 28: `vmatchPDict` implemented.
- Mar. 31: All other small tasks for `PDict` objects and string matching completed. Aim 2 complete; Biostrings release for Bioconductor v3.21 finalized.
- Apr. 30: Optimized `readXStringSet` function completed.
- May 30: Support for compression levels and other compression formats for `writeXStringSet` implemented. Aim 3 Complete. Codebase available in Bioconductor devel branch (v3.22). Funded portion of proposal ends.
- July: Present at Bioc2025 and useR! 2025.
- October 2025: Code finalized in Bioconductor release version 3.22.

Other Aspects

This project will be highlighted through a variety of methods.

Changes made will be released via Bioconductor's semiannual version releases. I will accompany these updates with social media outreach and blog posts to my website (pending approval to be cross-listed at R-bloggers.com).

As mentioned previously, this proposal will conclude in June 2025, shortly before the annual useR! and Bioconductor conferences. I will apply to present the work done in this proposal at both of these conferences. This will also provide a good opporunity to highlight the ISC's contribution to this project.

Requirements

The primary requirement for making this project happen is a developer with the time and willingness to maintain Biostrings, and the technical ability to make that happen. A secondary requirement is funding to support the developer during their work on this package.

People

I (Aidan Lakshman) will be the primary developer on this project. I am a PhD student in the Department of Biomedical Informatics at the University of Pittsburgh. In my work, I am a developer of the SynExtend package, which is dependent upon Biostrings. I have a high level of expertise with R, as demonstrated by my submissions to base R (e.g., dendrapply, wilcox) and my contributions to Biostrings (e.g., AAStrings). During my work on Biostrings, I developed a strong working relationship with the current maintainer, Hervé Pagès. If funded, I will conduct this project during the last year of my PhD, and then the package will be in a state where I can continue to support it longer-term in conjunction with others in the Bioconductor community.

Auxiliary supporters of this proposal are my PhD advisor, Erik Wright, and current Biostrings maintainer Hervé Pagès. Erik Wright is a past contributor to Biostrings, and is supportive of me committing 20% of my work hours to this project. Both Erik Wright and Hervé Pagès will provide advising throughout the project to ensure contributed code is high quality and to prepare me to become a long-term maintainer of Biostrings. Their primary contribution will take the form of code reviews and suggestions for additional improvements.

I have already met with both Erik Wright and Hervé Pagès to develop the structure of this proposal–letters of support are available at the top of this proposal.

Processes

No specific processes are required. The work done in this proposal will be regularly disseminated to the community at large via regularly scheduled Bioconductor releases (see "Measuring Success") and via conferences in summer 2025 (see "Other Aspects"). I will regularly meet with Hervé Pagès to ensure contributed code is of the quality expected for the Biostrings package.

Tools & Tech

No specific tools or technology are needed to deliver this project. The Biostrings codebase is hosted on GitHub and is released regularly on Bioconductor. All software development can be done on my personal computer.

Funding

I respectfully request $8,000 to cover my labor cost associated with the 20% effort for 1 year needed to complete this project. This amount is based on my current $40,000 per year stipend as a graduate student at the University of Pittsburgh.

Summary

The costs of this project are entirely labor hours. This project will take a substantial fraction of my workload to complete, and funding will support that effort. This work focuses on eliminating accumulated technical debt in Biostrings; at the conclusion of this project, the workload to maintain Biostrings will have been significantly reduced. In other words, the funding for this project will allow me to continue to maintain Biostrings in the future without further financial support.

Success

The overall goals of this project are to clean up the backlog of TODOs, make Biostrings easier to maintain, and transition future maintenance to a new owner. Success is thus not just in terms of short term contributions to the package, but also how the burden of maintenance is decreased for future developers.

Definition of Done

A successful outcome for this project is the following:

1. Biostrings is easily maintainable using automated testing suites.
2. All user-submitted bugs from prior to June 2024 are resolved.
3. Major items in the backlog of Biostrings TODOs are addressed.
4. Biostrings has the capability to handle big biological data.

Outcomes 1 and 2 result from Aim 1, Outcome 3 results from Aim 2, and Outcome 4 results from Aim 3.

Of these, Outcome 1 is arguably the most important. Biostrings is a critical package in the R ecosystem, and will undoubtedly persist for years to come. As a result, it is critical that the codebase is made as maintainable as possible, both for myself and for future maintainers. Code that is easier to maintain is easier to contribute to, and thus lowers the barrier to entry for community members to be involved with Biostrings.

Outcomes 2-4 will greatly improve user experience and improve the ability of R to handle larger and larger analyses. Given the deluge of biological data in the modern era, these improvements are essential for the continued success of R for bioinformatics.

Measuring Success

Success will be measured through well-defined milestones. Many of the specific deliverables are mentioned in the "Technical Delivery" section.

Each Aim is composed of a set of discrete tasks. We can thus measure progress by how many tasks have been completed. Biostrings is distributed through Bioconductor, which has two version releases per year. These line up perfectly with the planned timeline of this proposal. The conclusion of Aim 1 is measurable by an empty issue tracker on GitHub, and will be completed for Bioconductor's version 3.20 release in October. The conclusion of Aim 2 is measurable by how many tasks in the TODO file have been completed, and will

be completed for Bioconductor's version 3.21 release in April. These tasks will be itemized on the milestones page for the project's GitHub (see below). The final Aim will be completed by the end of the proposed timeline, in June.

If funded, I will add an itemized issue tracker to this proposal with percentage completion rates. This will be regularly updated with the project, allowing anyone interested in the project to follow its status. This project tracker is available on the proposal GitHub.

Future Work

I plan to continue maintenance of this project into the future. This funding will provide support for me to gain an intricate knowledge of the codebase and create a testing infrastructure, allowing future maintenance to be much faster and less of a development burden.

Future work falls into two categories:

1. Continued Maintenance. This is the expected burden of maintaining the package: dealing with bug reports, responding to users on GitHub and Bioconductor, and keeping the package updated for each subsequent Bioconductor and R version release.
2. Feature enhancements. This includes adding additional enhancements or features, including the various "nice to have" optimizations detailed in the Biostrings TODO file. An illustrative example of this is adding an internal `NucleotideString` virtual class to simplify the codebase. Many functions employ checks like:

```
if (is(x@subject, "DNAString") || is(x@subject, "RNAString")) ...

This change would simplify these to just:

if (is(x@subject, "NucleotideString")) ...

This would also simplify later unit testing. Enhancements like these are reserved for future work since
```

It's also important to emphasize that future work will target community involvement. Once the initial work of cleaning up the package and implementing solid testing is done, allowing and encouraging contribution will be much easier. A focus of the planned conference talks at useR! and BioC2025 will be trying to get more community members involved in contributing code to Biostrings.

Key Risks

Much of the technical scaffold for this project is already in place: it exists on GitHub, is distributed through a carefully curated package manager with regular release times, has an itemized list of tasks, and doesn't require additional tooling or technology to complete.

The primary mode of failure of this project is tasks taking longer than expected to complete. For this reason, tasks are arranged in order of decreasing importance: Aim 1 is the most critical, followed by Aim 2, and finally Aim 3. This project will allow me to subsume the position of maintainer of Biostrings–in the event that some tasks do not complete within the timeframe of the project, I will continue to work on them until they are completed.
