Turiba University

AJWIN JOJU VENNATTUPARAMBIL

FRIENDLY CODING

PROFESSIONAL BACHELOR THESIS

Study Programme: Computer Systems

Author: **AJWIN JOJU VENNATTUPARAMBIL**

***Thesis* Advisor:** **Janis Peksa**

Introduction

This report provides a warm and approachable overview of coding concepts, best practices, and ways to improve code quality. It's designed for learners, hobbyists, or teams who want a positive, collaborative tone while still sharing useful technical insights.

Friendly coding refers to the practice of writing software in a way that is readable, maintainable, and welcoming to both current developers and anyone who may work with the code in the future. It combines technical clarity with positive communication, emphasizing collaboration, empathy, and accessibility in software development.

2. What Is Friendly Coding?

Friendly coding is coding that:

- Is understandable to others
- Uses clear structure and consistent style
- Contains helpful comments and documentation
- Avoids unnecessary complexity
- Encourages teamwork and positive communication

It prioritizes people as much as technology.

3. Key Principles of Friendly Coding

3.1 Readability

Readable code saves time and reduces frustration. This includes:

- Descriptive variable and function names
- Clean formatting and indentation
- Logical structure

```python
# Good
def calculate_total_price(items):
    return sum(items)

# Not-friendly
def c(x):
    return sum(x)
```

3.2 Helpful Comments

Comments should clarify the *why*, not repeat the *what*.

They should be concise, positive, and informative.

Example:

```python
# We apply a discount if the user is a member
if user.is_member:
    apply_discount()
```

3.3 Simplicity

Friendly code avoids unnecessary complexity.

Simple code is easier to teach, understand, debug, and maintain.

3.4 Consistency

Using a consistent style across a project:

- Improves readability
- Makes collaboration easier
- Reduces misunderstandings

Style guides (like PEP 8 for Python) help ensure consistency.

3.5 Collaboration

Coding is rarely a solo activity. Friendly coding includes:

- Writing clear pull requests
- Giving constructive feedback
- Respecting teammates' skill levels
- Sharing solutions and knowledge

Positive communication strengthens team productivity.

---

4. Benefits of Friendly Coding

Friendly coding offers multiple advantages:

4.1 Better Maintainability

Future developers—including your future self—can easily understand and update the code.

4.2 Fewer Bugs

Clear and well-documented code makes errors easier to spot.

4.3 Faster Onboarding

New team members understand code structure quickly.

4.4 Improved Morale

A supportive coding style reduces stress and promotes a healthy team culture.

---

5. Friendly Coding in Practice

5.1 Documentation

Friendly projects include:

- Setup instructions
- Usage examples
- Clear explanations of key components

5.2 Testing

Well-named tests help ensure reliability:

```python
def test_login_fails_with_wrong_password():
    ...
```

5.3 Version Control Habits

Friendly coding includes:

- Writing meaningful commit messages
- Keeping branches organized
- Reviewing others' code respectfully

Conclusion

Friendly coding is more than just writing code—it is about creating an environment where developers feel supported, empowered, and understood. It increases productivity, reduces errors, and promotes a culture of shared success. Whether you're a beginner or a senior engineer, practicing friendly coding makes software development better for everyone.