# Instructions

GitHub link for all the programs: <https://github.com/crispyChristean/Software-Development-1000---Team-Project>

<https://westernwyoming.instructure.com/courses/16987/discussion_topics/185035>

Christian: 307-922-4674

Grant: 307-871-6057

Rylee: (307)203-6968

GitHub Link: [https://github.com/crispyChristean/Software-Development-1350---Team-Project](https://github.com/crispyChristean/Software-Development-1000---Team-Project)

Follow these steps carefully:

1. Assign team roles. Some examples include:

* 1. Team lead
  2. Developers (will likely need two as this will be the bulk of the document)
  3. Tester
  4. QA manager/engineer

2. As a team, write the coding standards mockup/template.

3. Submit the completed document. Be sure to include the "team" names and roles. The team leader should be the one to compile and submit the document. The document can be an .md (README), .txt, or even a Word document.

I found some examples on GitHub that might be useful:

* [https://github.com/axolo-co/developer-resources/tree/mainLinks to an external site.](https://github.com/axolo-co/developer-resources/tree/main)
* [https://github.com/devspace/awesome-github-templatesLinks to an external site.](https://github.com/devspace/awesome-github-templates)

Coding Standards

[Instructions 1](#_Toc912157242)

[Authors/Roles: 2](#_Toc1317085265)

[Introduction 2](#_Toc141830892)

[Common Commenting usages 2](#_Toc545713485)

[Example 2: Programming Comments on a modified program. 3](#_Toc9026210)

[Commenting on New Programs and Multi-user Programs 4](#_Toc719327369)

[Heading Content 4](#_Toc538866659)

[Body Content 5](#_Toc1930894415)

[Footer Content 5](#_Toc168813678)

[Variable Naming Conventions 5](#_Toc653338752)

[Function Naming Conventions 6](#_Toc1440325929)

[Constants Naming Conventions 7](#_Toc932311427)

[Indentation 7](#_Toc1340721474)

[Communication Section for this document 8](#_Toc1712979556)

## Authors/Roles:

Team Lead Grant Bossa

Developers Christian Espinoza Celis

Grant Bossa

Tester Rylee Leavitt

QA Manager/Engineer Wolfgang Meyer

GitHub link for all the programs: <https://github.com/crispyChristean/Software-Development-1000---Team-Project>

## Introduction

This document is intended to help the new person working in our programming location and is mainly directed towards the programming expectations regarding proper commenting in our programs.

Notes:

* Most of the examples of coding are done in python, some of the visual details and programming syntax may vary but will share the same ideas and concepts within all of them.
* Python, Java, JavaScript, C++ are examples of programming languages that follow coding standards and the standards we have mocked up in this document.

## Common Commenting usages

Below are some basic formatting rules for comments when a program is created and edited. First, there is an example program showing the proper comment format and usage discussed below. The second example shows the interaction between multiple programmers that have worked on a program. Please note the difference between them.

Example 1: Programming Comments on the initial program.

#

# Grant Bossa

# Sept 18, 2024

# Budget Analysis Programming Project

# COSC 1010 NT

#

# Instructions:

# Write a program that asks the user to enter the amount that he or she has budgeted for a month.

# A loop should then prompt the user to enter each of his or her expenses for the month and keep a running total.

# When the loop finishes, the program should display the amount that the user is over or under budget.

# Use comments liberally throughout the program.

# Declare Variables:

budget = 0   # budget amount

monthly\_expense = 0   # total of expenses for the month

running\_total = 0   # running total of expenses

keep\_going = True   # loop control variable

# asks the user to enter the amount that he or she has budgeted for a month

budget = float(input("Enter your monthly budget: "))

# loop

# enter expenses for the month

while keep\_going:

    monthly\_expense = float(input("Enter your monthly expense: "))

    # keep a running total

    running\_total += monthly\_expense

    # ask the user if they want to enter another expense

    # if the user doesn't want to enter another expense, break the loop

    keep\_going = input("Do you want to continue? (y/n): ").lower() == "y"

# end loop

# display the amount that the user is over or under budget

if running\_total > budget:

    print(f"You are over budget by ${running\_total - budget: .2f}.")

elif running\_total < budget:

print(f"You are under budget by ${ budget - running\_total: .2f}.")

else:

    print("You are exactly on budget.")

# end of program

This first example shows the creation of a program by a single user. Please note that the comments made by the initial programmer are devoid of any information other than the purpose of the comment.

### Example 2: Programming Comments on a modified program.

#

# Rylee Leavitt

# Sept 23, 2024

# Budget Analysis Programming Project

# COSC 1010 NT

#

# Instructions:

# Make changes so that monthly budget becomes weekly budget

#

# # # # # # # #

#

# Rylee Leavitt

# Sept 21, 2024

# Budget Analysis Programming Project

# COSC 1010 NT

#

# Instructions:

# Make statements more consistent

#

# # # # # # # #

#

# Grant Bossa

# Sept 18, 2024

# Budget Analysis Programming Project

# COSC 1010 NT

#

# Instructions:

# Write a program that asks the user to enter the amount that he or she has budgeted for a month.

# A loop should then prompt the user to enter each of his or her expenses for the month and

# keep a running total.

# When the loop finishes, the program should display the amount that the user is over or under budget.

#

# Use comments liberally throughout the program.

# Declare Variables:

budget = 0   # budget amount

# 9/23/24 RL: Change the monthly\_expense to weekly\_expense

weekly\_expense = 0   # total of expenses for the week

running\_total = 0   # running total of expenses

keep\_going = True   # loop control variable

# asks the user to enter the amount that he or she has budgeted for a week

# 9/23/24 RL: Change the user input from monthly budget to weekly budget

budget = float(input("Enter your Weekly budget: "))

# loop

# Enter expenses for the week

while keep\_going:

# 9/23/24 RL: Change the monthly\_expense to weekly\_expense

    weekly\_expense = float(input("Enter your weekly expense: "))

# keep a running total

# 9/23/24 RL: Change the running total += monthly\_expense to running total += weekly\_expense

    running\_total += weekly\_expense

    # ask the user if they want to enter another expense

    # if the user doesn't want to enter another expense, break the loop

    keep\_going = input("Do you want to continue? (y/n): ").lower() == "y"

# end loop

# display the amount that the user is over or under budget

if running\_total > budget:

  print(f"You are over budget by ${running\_total - budget: .2f}.")

elif running\_total < budget:

    # 9/21/24 RL: change: ${budget – running\_total: .2f}.”) to ${running\_total – budget: .2f}.”)

print(f"You are under budget by ${running\_total - budget: .2f}.")

else:

    print("You are exactly on budget.")

# end of program

Please note that the comments made by the following programmers contain additional information other than just the purpose of the comment. They also add the date of the modification and the programmer’s initials to indicate when the changes were made and by whom they were made.

### Commenting on New Programs and Multi-user Programs

#### Heading Content

1. The header should contain the name of the programmer (If the program is multi-user, there will be more than one programmer as shown in Example 2), the date of the program changes; the most recent changes being nearest to the top, what changes were made in the form of instructions, the title, and the class identifier
2. There should be a section that describes the overall purpose of the program.
3. There should be a section that shows the Constant and Declared variables used in the program.
4. The variables should have inline comments, Inline comments should include the date of change, initials of the editor (If in a multi-user program), and a short description of the change that was made, telling what the statements reference or why they are being used. Inline comments should also be before the executing statement. Tab inline comments so that they are in alignment with the associated statement, this is to help maintain readability.

#### Body Content

1. Keep the line placement in line with the code block. If you are inside a loop, maintain the loop format for adding comments. If the statements are nested, then the comments need to be as well.
2. As stated previously, a multi-user program that uses inline comments needs to include the date and programmers’ initials (showing who is responsible for the change and when.)

#### Footer Content

1. The Footer Comment follows the last executable code of the body and indicates the end of the program.
2. In some instances, there may need to be processes that are completed at this stage of the program and they will have their code and comments below.

## Variable Naming Conventions

* Variables can be assigned names to help identify what data it is holding/referencing and what is associated with that variable. There are mainly two different types of variables. While there are different types, both must follow similar naming conventions to keep in mind.
* Global Variables - are a type of variable that is universally apparent and universally used across the whole file/folder.
* Local Variables - are a type of variable that is only usable in certain sections of the program. Such as a function within a program.

# Christian Celis

# 09/20/2024

# SDEV 1000 NT

# Example for Variable Naming Schemes

# Code is done in python language.

# This program is an example of Naming Conventions for our Coding Standards, both are acceptable because

# of legacy programming.

# This variable displays an example of Camel Casing, which shows the variable written with

# each word being used being capitalized at the start to read the name.

variableExample = 0 # example variable

# This variable displays an example of Snake Casing, which uses the underscore

# to represent different words.

variable\_example = 0 # example variable

General Notes about Variable Naming Conventions:

* Avoid using too many words for your variables, keep it easy to call.
* Avoid using single letters or characters when naming a variable as more detail will help avoid causing issue within the program.
* Avoid using only numbers or characters to define a variable as it may conflict with the programming.

## Function Naming Conventions

Generally, function naming conventions follow the same idea of Variable Naming Conventions, with the notion that they’re for functions, which is usually a set of statements or programming instructions to execute a certain task within a program.

# Christian C

# 09/23/2024

# SDEV 1000 NT

# Program example for function naming conventions

variableExample = "0" # example variable

# Function Names usually contain no keywords, no spaces, and a letter as the first character

# Function names carry similar rules to variable names.

def functionExample(): # This is an example of a function name, done in Camel Casing.

# The following statements are what are usually included in functions.

variableExample = input("What is the text you would like to display? ")

print(variableExample)

# This statement calls the function via the function name.

functionExample()

General Notes about Function Naming Conventions:

* Cannot include a programming language’s keywords.
* Avoid using spaces.
* Always try to use a letter as the first character for the name. Using numbers and symbols as the first breaks conventions.

## Constants Naming Conventions

Constants in programming are usually variables that have associated values that never change in the program. They can differentiate from variable naming schemes as they are different to help represent the difference between the two.

# Christian Espinoza

# 09/22/2024

# SDEV 1000 NT

# Constants Naming Conventions Python example.

# Constants are variables that don't ever change value

CONSTANT\_EXAMPLE = 0 # This is an example of a variable constant.

# Constants are usually capitalized in the format shown but may vary.

# They are completely capitalized to help distinguish them from other variables.

* Conventions can vary with each programming language. Usually, they will always capitalize and have no spacing.

## Indentation

Indentation is a common practice in Programming and is often, required to do so for the program to function. The purpose of indentation is to allow the programmer/tester to better view aspects of the code, this includes functions, if statements, and other programming statements. This is especially necessary for making complex programs that utilize many concepts within (such as an if statement within an if statement or a while statement in an if statement.

# Christian C

#09/23/2024

# SDEV 1000 NT

# This program example helps clarify the use of Indentation in python.

#Variables declared

firstVariable = 2 # example variable

secondVariable = 3 # example variable

# Here is an example of Indentation, the if statement helps show a possible of sequence of events

# to follow

# Within the if statement.

if firstVariable > secondVariable:

# Indentation here indicates everything within the if statement.

print("The first variable is not greater than the second")

else: # Indentation here indicates another possible sequence of events.

print("The first variable is greater than the second")

* It is important to keep in mind that indentation, and when to indent, can differentiate between each programming language. Nonetheless each has a standard.

# Communication Section for this document

**This area will be used for communication if more than 1 person is using the document**

CC: How are you producing the tables, may I ask? The one with the green text.

Gb: For the table of contents, it is a toggled field code that updates when you right click on it.

Gb: The text above was a copy / paste from visual STUDIO

Gb: Have you noticed on the top of this display in the toolbox section, it should have a button with my initials. This will help in being able to have us both in the document at the same time. I am glad this works.

CC: I see, I will use visual studio then just paste my contents once I’m done. My variables and content I produce will be made in Python as that is the current language that is set for my visual studio, which should match your code entirely.

CC: I think we should also keep our conversations on this word document as it shows we are communicating on the project. Also yes, I have noticed the initials, and it has helped in showing your additions!

Gb: I will ask her about this, I am sure that as a final document, we will have to delete them but should be good for devel

CC: Hmmm, by that then it could make sense that we insert the comments we make here onto a new document separate from this one to still show we had conversations but does not interfere with the final document. Of course, once we get this project submitted.

Gb: I think we can submit twice. One with comments and the other without. That way we have proof of collaboration but have a finished document as well.

Gb: One heads up would be to use our initials at the start of the comment as well and I am really bad about not doing that.

Gb: I am changing the code display in this doc, to a size 8 font. Let me know if it looks better or not and we can review closer to being finished to clarify and standardize.

CC: No worries, I will try to be careful placing my initials in each comment. Also, I think the code displays work well and considering the amount of content we’ll add will help reduce size.

Gb: I also changed its paragraph size to 10 so it isn’t too large. I would like to get

CC: I just submitted a code example for the variable naming, what do you think? This is how most of my examples will look like if it fits our needs well.

CC: I see your comment on the code example, I will try to align that better. If it overall works, then I will be using somewhat of the same format moving forward.

Gb: Comment first, then the variable, watch your spacing on the carryover lines and add an inline comment on the variable definition

CC: I agree, I think this project will turn out well and we have plenty of time in the next few days to get large quantities done. I will head off for the night but as I explained I'll try to work on this more tomorrow! Same for you as well! Hope you rest as well.

Gb: that works great. I think we’re on to a late start but will be turning in a great product

Gb: I think this would be great. I hope you have a good night and get some rest.

RL: Hello, I am super sorry I am behind in the class as of right now. My friend was in an emergency type situation for a week (give or take) that required immediate attention. Please harass me as I can lose track of things sometimes. My phone number is (307)203-6968 and my email is [ryleeleavitt@westernwyoming.edu](mailto:ryleeleavitt@westernwyoming.edu). I see I was assigned as a Program Tester; however, I can perform any task the team needs, please let me know when I can be expected to do these tasks. Thank you so much! I did not intend to leave you all in the dark, documents do not have notifications XD

RL: My cell phone is having difficulty with phone calls, please schedule calls with me and call my partner Tanner R. at: (307)231-9032

RL: (Phone call with Gb, allotted time: 52:09, 9/21/24) discussion/instruction

RL: I may have accidentally added a line in the statement (Program change comment): I was trying to show my change to the code in the pursuit of maintaining consistency of functions/statements. Please let me know how this will affect submission.

if running\_total > budget:

    print(f"You are over budget by ${running\_total - budget: .2f}.")

elif running\_total < budget:

    print(f"You are under budget by ${running\_total - budget: .2f}.")

# 9/21/24 RL: change: ${budget - running\_total: .2f}.") to ${running\_total - budget: .2f}.")

RL: I also updated Heading Content, Body Content, and Footer Content (9/21/24) under the section “Commenting on a Multi-User Program.” Adding that “Program changes should be identified via comments including the date of change, initials of the editor, a description of the change that was made, and finally the title, and the class identifier” and “Any statements/Functions should be parallel and consistent in structure/format.” I will be making sure statements/functions are parallel to one another on top of testing. GB, please let me know if my changes were satisfactory. 😊

RL: I approve of GB being our team lead as he demonstrated to me his experience via our phone conversation.

Gb: Rylee, a couple things to remember. Consistency (1), information (2), and modifications (3). 1. Why is the comment after the change you made? You normally want to comment before the executable. 1. Keeping the comments in the structure of the statements they are in… you commented on an indented line, the comment would be indented as well to maintain structure. 2. Where is your programming information in the header? 3. Modifications to the document don’t have to be dated but you should keep an eye on the numbering of the items in a list. All three sections have wrong numbering. You should be able to right-click and restart numbering

CC: Rylee, I see that you joined, I will try to contact you here soon, I will try to add some more content on the document for it to get ready for tomorrow. Also, since I never clarified who the team lead should be in text format, I more than approve of Grant of being in the position of a team lead.

CC: I have a small list of topics I will add, this includes the Naming Conventions for Functions, Naming Conventions for Constants, and Programming Indentation. I think once we get this done, we should add some indentation to the document itself to make it easier to read.

RL: Grant, thank you for letting me know, I will make these changes. As for the code in section “Commenting on a new program by a single user.” does this code have a break statement that will trigger the end of the program. I see you have *“keep\_going = input("Do you want to continue? (y/n): ").lower() == "y"* will this statement trigger given that the loop control variable is *keep\_going = True.* In COSC 1010 to get this function to trigger I had to add a break statement as well as *keep\_going* did not equal true. I attempted to run the program and the user input to trigger the end of the loop did not process. Of course, the monthly budget program I did for COSC 1010 is no longer in my files.

RL: Grant, never mind the code does in fact run correctly in visual studio. I was simply not using the correct extension. As you have more experience than me in code, outside of this project of course, would you mind explaining why you did not have to use a break statement? Thank you! 😊 also, there were periods in the code when I copy and pasted it. I am sure that this was simply for readability purposes, but it did issue an error in the terminal when it ran.

Gb:It triggers because if you enter anything but a ‘Y’ or ‘y’ the value becomes false and the while loop then doesnt run anymore. The next statement after this input is the test for the while loop.

RL: Grant, I changed the comment I made in the program to be structured similarly. As for my programming information in the header, could you please give me an example? Do you mean placing my information by yours? for example:

Grant Bossa

# Sept 18, 2024

# Budget Analysis Programming Project

# COSC 1010 NT

Rylee Leavitt

# Sept 22, 2024

# Budget Analysis Programming Project

# COSC 1010 NT

I apologize for my confusion. I also fixed the numbering in the Heading, Body, and Footer content (9/21/24) lists as you suggested.

Gb: Rylee, newest header comment to the top, including a programming changes information statement. That way the newest changes are listed at the top. Kind of like a program update newest posting to the top so that the newest changes can be easily found. The originator would be at the bottom of the list.

RL: Christian, thank you for reaching out to me. I appreciate your help and understanding. I Have noticed your addition of topics to the document, I think these changes are beneficial to the overall structure and understandability for a new team member. Variable and function naming conventions can be tricky, so I appreciate the addition to the document. Let me know if you need anything from me. Thank you!

CC: Of course! I’m just looking back again at the added content and wow you guys have been adding a lot! With the footer, header, Commenting! It’s all a lot! I’ll be adding the program examples for naming conventions tomorrow, but it shouldn’t take too long at all! I’m looking at some of my resources and I think I will add some more content that will go over the UTF-8 standard character sets, data types, and indentation tomorrow, I will be unavailable between 6am-2pm tomorrow but I still be good to answer for text and will be able to finish the bulk of the items I just listed into the document around 5pm. Which aligns well with Grant’s time to potentially submit the assignment.

RL: I will be unavailable between 11-12 tomorrow as I have a meeting in the peer tutor center, I was able to answer my own questions, but it'll still be good to get help with some concepts 😊 I will be available through text as always. But everything on my end should be ready by 5.

GB (Text Msg): Rylee, look at the commenting on the single vs the headers under it and the same for the second example? 1 subheading under the multiuser. The header levels are different. They should stagger

H2,

H3,

H4

The body and footer got changed to regular text that’s why they don't show up in the table of contents.

RL (Text Msg): Grant, I see at the beginning of the doc the header goes from h2 immediately to h4. What should h3 be between common commenting on a new program by a single user?

GB (Text Msg): yes, the h4 should be h3 the header/body/footer should be h4.

RL (Text Msg): Grant, how do we do h2 before commenting on a multiuser program?

GB (Text Msg): Common commenting usages is the h3 above them. In the comment under common it states there will be 2 examples. Now that is reflected in the Table of Contents as well. Would you please read both the single and the multiuser information and make sure that we reference one without it being a copy?

RL (Text Msg): I think we are ok there, that doesn't feel like a copy.

GB (Text Msg): I've bolded some duplicate info; the second example should refer to the first in that the directions are the same. Additions, adding the date and initials should be noted for changes as well as addition of a second hdr to the second example. Word tracks who makes the changes and when. It has its own revision history. Especially when it comes to multiusers because what happens if erases entire pages?

GB (Text Msg): Rylee, please add your information to the copied info on the second example. You were correct before by having the line comment before the changed statement. However, you also changed the text in the original pgm. The initial pgm wouldn't have changes as it was the original. We are trying to show the distinction of the original pgm and the changes that need to be made in the second.

WM I have been reading through this document a lot and it looks really good. I will offer any suggestions if I see an area that needs improvement, but I really think that this document looks great and has all the necessary information and styling!

CC: Hey Wolf, can you share your number with the rest of us on the top of the document? That way we have another way to communicate if need be!

CC: Hello, I am just getting back into this document. I’m about to implement programming examples for naming conventions and go from there. I see that you have finally accessed the document wolfpack, It’s nice to see that you were able to reach back!

RL: Hey guys! Do you need anything else from me? All the code looks good, runs well, and doc is updated on my end 😊

CC: I’m not actually sure! I guess if the doc is updated and stuff, I guess it seems that all I got to do is pull my own weight with the stuff! I am adding the python examples in constants and functions right now and maybe checking if I plug those program examples in correctly in the doc (in terms of formatting) could help!

RL: Okiedoke

CC: What does everyone think about plugging in the code examples into the git I made in github for this assignment? It would just contain the files I have that contain the actual examples I have written as well as everyone else's! Since the document only shows the code in text rather than the actual file.

RL: I think that is a good Idea! That way we will also have a backup. That way we could also download them into our own repositories and use it for future reference. Also, Grant is trying to log in to the doc, I think the internet is tricky. Grant suggested (over text) that we should combine the directions of both sets of Header/Body/Footer content into 1.

CC: I think that would actually help relieve the amount of content in the doc in a way (as in it will help make it look more organize)! I think it’s a good idea that could be a quick and easy change!!!! I am going to put the link in for the git on the top of the document and start adding the programs I made, with the link feel free to add your guy's programs as well! Also I just got done writing the example naming conventions for constants and functions!

RL: I'll get started on that then! Thank you Christian.

GB: Stold your mouse. Sorry

CC: Grant, do you have any recommendations on other coding standards we should add real quick? Like indentation in coding?

Gb: Let me look over what we have already. I sent an e-mail regarding Dr. T’s comment about the GitHub as well as the 2 copies of the document. She liked the idea. We need to give her an address for the Github as well as the 2 documents.

RL: Christian put the link to the repository above and what if we submitted the docs to the repository as well?

CC: I like the idea of making a copy and uploading it into the git in case something happens, or we need to reference an older version, is it alright if I make the copy once we have finalized everything?

RL: I think that’s fine! Thanks Christian!

CC: Awesome! Just let me know through text or here once we are ready for me to do so!!

CC: Hey, you guys I noticed that Dr.T has made a post in the discussion canvas!!

RL: We don't need an extension, do we? I thought we were really close to submission?

CC: That’s what I thought as well, I wanted Dr. T to see our work and judge whether or not we need an extension since I thought we were close as well.

RL: What do you think Grant? I think that's a good idea Christian. Let's send it to Dr. T and she could decide if we need to add more content. When we are ready for review of course.

CC: It shows that grant is out of the document, I’ll text him really quick if he noticed Dr. T’s response. Okay mine shows that now too.

RL: I'm going to go eat dinner real quick, text me if you need me, you guys already have my number

CC: Understood! I am still trying to contact Dr. T about this, she has stated we come at our own terms on whether or not we need an extension and have tried to be vocal about our progress as a whole in this.

GB: Rylee do you think we need an extension?

CC: Hey Rylee, currently both Grant and I are on a call discussing submission details. From the sound of it seems like we are good.

Gb: Thank you, I'm going to contact Dr. T. and make sure that we follow her format

**GB: Please respond to Dr. T’s discussion as far as the need to have an extension.please respond here as well so that we can submit**

RL: I think we are good for submission

WM I have been reading through this document a lot and it looks really good. I will offer any suggestions if I see an area that needs improvement, but I really think that this document looks great and has all the necessary information and styling!

GB: I think it is ready to submit as well

RL: Grant, you are the one to submit the document, right?

GB Yes, I will copy the document to the same collab folder as this one, I will add final to the title and will wait until CC gets it copied to github.GitHub Before submitting it to Dr. T.

CC: Besides some finalizing we are doing once we submit; I believe we are good to submit!

CC: Grant has communicated with me, here is how things will be submitted from my understanding. Right now, I’m going to download a copy of this document and post it on the GitHub shared, once grant makes the final version (the versions that excludes the conversation into its own document) he will send me those and I will upload the final versions onto github as well. Since Grant is the team lead the team lead is the only person who needs to submit from my understanding (lol). I just want to make sure everyone is on the same page about how things will be submitted. I will also make a group chat in text detailing that I have made the submissions on GitHub that way everyone knows when the document is submitted.

RL: Ok, thank you Christian!

CC: YEeAh! I’m glad we got the documented (coding standard mockup) getting submitted and worked on! Thank you Rylee for the help!!!

RL: YeEaAhH! Absolutely, Thank you! This project went pretty well. I look forward to maybe working together again in the future.

CC: Okay I have just submitted the draft on github and am just waiting for the finalized versions! Also, I must agree, this went well, and I look forward to any collaboration work in the possible future.

GB: Ready to upload File