

# Programming Language Research Project

## Introduction

The course is designed to give you the necessary skills to evaluate and learn new languages effectively. This is an important task for any computer scientist, and it will give you experience in applying the concepts of this course.

You will be working in groups of 2 (a group of 3 will only be permitted if an uneven number of students exist in the class i.e. at most there will only be one group of 3). Each group will be assigned a language to research, write about, and present to the class. I will decide what language each group is responsible for based on the preferences of the group, and the goals of the course. The presentations will be done during the last week of the semester.

## Schedule

In most cases, there is nothing to submit to me for these deadlines. These deadlines are guidelines for you to complete your project in a timely manner. The WebCT submission sites will remain open until the end of the semester for the installation screen shot, first draft report, final report, and PowerPoint presentation. You will not get grades on each of these, but your timeliness will be reflected in your final grade.

**September 12<sup>th</sup>:** **Submit** names of students in the group via e-mail.

**September 26<sup>th</sup>:** **Submit** list of 5 preferred languages (one list per group via e-mail).

**October 3<sup>rd</sup>:** Notification of group's assigned languages (discussion group on WebCT)

**October 17<sup>th</sup>:** Installation of a compiler/interpreter should be completed by this time. **Submit** a screen shot and code listing of the sample program from the appendix to WebCT to show that this has been done.

**November 3<sup>rd</sup>:** **Submit** first draft of report. No submission is required, but I will be happy to review your report. This gives you 3+ weeks to finalize the report.

**November 21<sup>st</sup>:** Schedule for presentations will be determined by random drawing.

**November 24<sup>th</sup>:** **Submit** written report is due in ACM formatting standard (note this is right before Thanksgiving). Report must be **submitted** in Word or PDF format.

**December 1<sup>st</sup>:** Presentations will begin. Your presentation must be **submitted** in **PPT or PDF format**.

## Choosing a language

Each group will report on only one language; however, each group will need to determine 5 languages that interest you and submit them. The languages can be

chosen from the following list:

- Ada
- Dylan
- Eiffel
- Haskell
- Icon
- JavaScript
- Oberon or Modula-2 or Modula-3
- Postscript or Forth
- Python
- Smalltalk
- SML
- tcl
- TeX
- Any Unix Shell
- Awk
- A visually-oriented programming language(with approval)
- SQL
- VRML
- XML
- Other (another language that is approved by the instructor)

This is not an exhaustive list of languages, but it does represent a nice cross-section of the varying types of languages. In addition, the interpreter/compiler availability (freeware) are typically readily available. You may choose a language that is not on this list, but see me about any language that is not on this list before you make one of your 5 languages (note the compiler/interpreter needs to be available).

Be advised that some of these languages are probably available either on our server, my machine (bubble.spelman.edu), or under the Linux OS (if you have this installed on your machine). You may use any implementation of the language that you choose. This means that you can use implementations for Windows, Linux, Solaris, MacOS, etc. Please see me for help with installations on various platforms. If you determine that the installation must happen on Solaris then you must see me early to have our Unix system administrator assist with the installation.

## **Submitting Your Language Preferences**

As indicated in the schedule, you will need to submit a list of preferred languages for

you group. To improve your chances of not conflicting with other groups, you may want to choose a language that is not in the list above. You may need to do some preliminary research on the languages you choose to increase your interest in the language. This is especially needed when you are choosing a language that is not on the list because you have to bring that information to me to verify that you can have that language on your list.

You should submit a single e-mail message for your group that includes:

1. A list of all group members
2. A list of 5 languages in order of preference.
3. Groups will be assigned languages based on how you rank them, and how other groups rank them.

## Researching the language

The reports will have two major parts: research of the assigned language and writing some code in your assigned language. The report should be no greater than 5 pages long and in the ACM standard 2-column format. I will not be grading based on length, but instead on quality. In other words, be thorough in your research; and if that takes 3 pages then stop at 3. If you find that you want to write a longer report to get the information reported, then you need to rethink the organization and content. You may also see me about ways to make your writing more concise.

You may use any resources at the Woodruff Library, other university libraries, the WWW (the class webpage has a few links to help you get started), and you can possibly find people outside of the class that can help you with information. The following is an outline that you can use to help you organize your reports. You do not have to follow this outline to the letter, but your report should be well-organized and discuss these topics.

### Background

- Who invented the language and for what purpose?
- What category of languages does your language fall into?
- What is the genealogy of your language? What did it use from its predecessors? What did it add? What did it leave out?
- Has your language evolved over the years with new features? New libraries? Etc.

### Basic Properties

- Overview of the language
- Programming Model/Computation Model
- Evaluation Order
- Name Spaces

- Data Model
- Parameter Passing
- Computation Organization (control-flow structures, functions, procedures, etc.)
- Type System (scalar types, user-defined types, objects, etc.)
- Notation: expressivity? Readability? Writability? Orthogonality? Etc.
- Evaluate the language according to the criteria and factors discussed in this course

### **Special Properties**

- Anything special that you find interesting in the language
- What can this language do that others can not?
- What kinds of applications is your language suitable?
- What is the language not suitable?

Your reports should not quote these headings and topics directly. You should organize your report to cover these and other topics. The lists above are only meant to be guidelines on the types of information that should be considered. You would have learned a number of language concepts in this course, and the report should show your knowledge and your ability to apply that knowledge on an unfamiliar language. Think of this report as something you might write if your supervisor asked you to evaluate the usefulness of a language.

## **A Program for Everyone**

The second part of your assignment involves writing some code in your assigned language. To make for some interesting comparisons, and to make it easier for everyone to understand the presentations, every group will create the same program. The sample program is found in the **appendix** as a C++ program using classes.

The 2<sup>nd</sup> program is one that you find interesting, and is motivated by the application domain of the language. This means that if the language is catering to mathematics then this example should be based there or if the language is targeted towards graphics then the example should be a graphics program. This program **must** be on the level of the size/complexity of programs you have written in this course at the level of your 2<sup>nd</sup> programming course. If you are unsure if the complexity is appropriate then see the instructor.

## **Deliverables**

### 1. A written report of your research (50%)

The paper and the presentation will be graded on quality and substance. You do not need to write everything about your language. Your paper should flow smoothly from one topic to another. You will be graded on the flow of your report, and on

how well you select and compact the information for easy consumption by the reader.

## 2. The code and results of your sample programs (20%)

You will be required to show me that your code runs i.e. A **demo**. You will need to provide a hard copy of the **programs** and the **outputs**, as well as **instructions** on how to execute the programs in the given environment. Your instructions should include the machine, OS, where the implementation was obtained from, etc. ***This is not a part of your report page counts.***

## 3. The presentation made in class (30%)

The presentation will be from 15-20 minutes. This includes a 5 minute question and answer period. You will need to include highlights from your report (not every word or idea), and the sample programs.

The presentation should be divided between the group members. You may divide this in any reasonable fashion.

**NOTE:** Everyone is required to attend all presentations. If you are not at a presentation, then your work will be deducted **5%** for each presentation that is missed. There will be multiple presentations per period.

**NOTE:** Timeliness of submissions is factored in as 10% of the grade. On time for all milestones means 10% increase on your grade for timeliness, and the fraction of late submissions will be used to determine how much of the 10% will be used for deductions.

## Appendix

```
// This program shows how an array based list would be created using
// C++ classes. The list of integers is a type that can be used to
// create objects. The operations are:
// 1. append an element
// 2. remove first element
// 3. sort the list
// 4. find element in the list
// 5. list comparison
// 6. length of the list
// 7. display the list
```

```
#include <iostream>
```

```
using namespace std;
```

```
class intList {
private:
    int* theList;
    int currentSize;
    int maxSize;
public:
    // constructor
```

```

    intList();
    // destructor
    ~intList();

    // operations
    void append(int value);
    int removefirst();
    void sort();
    bool find(int value);
    bool operator==(const intList& rhs);
    int length();
    void display();
};

intList::intList() {
    theList = new int[10];
    maxSize = 10;
    currentSize = 0;
}

intList::~intList() {
    delete theList;
}

void intList::append(int value) {
    if (currentSize == maxSize) {
        maxSize *= 2;
        int* newList = new int[maxSize];

        // copy old list to new list
        for (int i = 0; i < currentSize; i++)
            newList[i] = theList[i];

        theList = newList;
    }

    currentSize++;
    theList[currentSize-1] = value;
}

int intList::removefirst() {
    int first = theList[0];

    // shift all values to the left by 1
    for (int i = 1; i < currentSize; i++)
        theList[i-1] = theList[i];

    currentSize--;

    return first;
}

int intList::length() {
    return currentSize;
}

```

```

}

bool intList::find(int value) {
    int i = 0;

    while ( i < currentSize ) {
        if (theList[i] == value)
            return true;
        i++;
    }

    return false;
}

void intList::display() {
    int i = 0;

    while ( i < currentSize ) {
        cout << i << ": " << theList[i] << endl;
        i++;
    }
}

bool intList::operator==(const intList& rhs) {
    int i = 0;

    while(i < currentSize) {
        if (theList[i] != rhs.theList[i])
            return false;
        i++;
    }

    return true;
}

void intList::sort() {
    for (int i = 0; i < currentSize; i++)
        for (int j = i+1; j < currentSize; j++)
            if (theList[i] > theList[j]) {
                int temp = theList[i];
                theList[i] = theList[j];
                theList[j] = temp;
            }
}

int main() {

    intList list1;

    list1.append(9);
    list1.append(2);
    list1.append(5);
}

```

```

list1.append(1);

cout << "list1 has the following contents: " << endl;
list1.display();

cout << "Sort the list1" << endl;

list1.sort();

list1.display();

cout << "remove first from list1" << endl;
cout << "removed: " << list1.removefirst() << endl;

list1.display();

intList list2;

list2.append(2);
list2.append(5);
list2.append(9);

cout << "the contents of list2: " << endl;
list2.display();

cout << "compare list1 and list2" << endl;
string result = (list1 == list2) ? "the same" : "different";

cout << result << endl;

cout << "now delete one from list2 and compare to list1" << endl;
cout << "removed: " << list2.removefirst() << endl;
result = (list1 == list2) ? "the same" : "different";

cout << result << endl;

cout << "list1 is length: " << list1.length() << endl;
cout << "list2 is length: " << list2.length() << endl;

return 0;
}

```