Modern Programming Languages (CS508)
SPRING 2012
Solution Assignment No. 1

Total Marks: 20 **Dead line: April 13, 2012**

Application:

AlstraSoft E-Friends

Scenario:

Suppose you are developing an online first-degree friendship building social network for **E-friends** application at **Alstrasoft Pvt Ltd** that strive to be the premier provider of web-based software solutions for businesses worldwide. As a web development and software company, Alstrasoft specialize mainly in web design and software programming.

Evaluation:

Now being a senior Software Developer at this company where portability is the prime concern of your management, you are required to answer following questions keeping in view the above constraints.

a) Which programming language among **Java and C++** will be preferred by you in this programming paradigm keeping portability a major concern for your company? State your answer with solid reasons? **[5]**

Solution:

Being a senior software developer at 'Alstrasoft' the language of choice for this E-friends application development would be Java. As java is more portable than C++. **The concept of Write-once-run-anywhere (known as the Platform independent) is one of the important key feature of java language that makes java as the most powerful language.**

Any Programming language is considered standardized if you can port it on any compiler that imply same standards. Certain language is difficult to port from one compiler to another because the compilers for those languages are not compatible with each other.

Java will be the language of choice for programmers in developing environments where portability is the major concern. Following are two basic reasons that java is considered more portable than C++:

1-JAVA is an **architecture neutral** language. Being architecture neutral is a big chunk of being portable, but there's more to it than that. Unlike C and C++, there are no "implementation dependent" aspects of the specification. The sizes of the primitive data types are specified, as is the behavior of arithmetic on them. For example, "int" always means a signed two's complement 32 bit integer, and "float" always means a 32-bit IEEE 754 floating point number. on the other hand C++ Data type size is not guaranteed. Integer can be of 2 bytes 4 bytes or 8 bytes but in java data types are standardized. The libraries that are a part of the system define portable interfaces.

2-Concept of JVM(java virtual machines) make this language more portable as compared to C++.These Virtual Machine codes, or JAVA byte codes, are all architecturally neutral; the compiled codes are executable on any processors and systems, provided of course, that the JAVA Virtual machine (interpreter) is installed in those systems. So, basically, the compiler generates bytecode instructions which have nothing to do with particular computer architecture. Rather, they are designed to be both easy to interpret on any machine and easily translated into native code.

b) Beside portability what other **factors** should be considered for language of choice in above scenario, to give better productivity in e-Business and to enable customers to start their own online business with ease and generate an extra stream of revenue. **[10]**

Solution:

Following are the factors that should be considered for language of choice beside portability:

- **Simplicity/orthogonality** – It is one of the vital factor that should be considered while opting any programming language for any project.it should be kept in mind that how simple is that language to program into and how easy it is to understand it for other developers too. a complex language will take more time and resources. Java simplifies arrays and pointers. Java has orthogonal I/O since all I/O is through classes. Further, formatted I/O can be simpler than in C++ if you use the proper Java class.

- **Control structures** – programming is usually not limited to a linear sequence of instructions. During its process it may bifurcate, repeat code or take decisions. For that purpose, every language provides control structures that serve to specify what has to be done by our program, when and under which circumstances.

  With the introduction of control structures we are going to have to introduce a new concept: the compound-statement or block. so for choosing any language we should consider this factor too.

- **Data types/structures** – For language of choice one should also consider that which data types that language can support. more the data types a language support better it is to program in. like if a certain language is object oriented then its is better to use that language as data abstractions would be there.

- **Syntax design** –It should also be simpler for any language to be chosen.

- **Support for Abstraction** –language should have excellent support – ability to define subroutines and data structures. It should permit separate compilation of code as well. I give the edge here to Java though because it is object-oriented.

- **Expressitivity** –A language should be expressive too such as through short cut operators and the flexible for loop. Because Java has objects, it is easier to express many concepts in Java.

- **Type Checking** – strongly typed language should be preferred .
- **Exception Handling** –A language should have the ability to handle all exceptions In general, an exception is handled (resolved) by saving the current state of execution in a predefined place and switching the execution to a specific subroutine an exception handler.

c) While working in same programming environment you came across following piece of code implemented in **C++**.from the following code infer if it is following the principle of orthogonality or not? Justify your answer. **[5]**

```
#include <iostream.h>
#include <stdio.h>
# include <conio.h>
 using namespace std;

int main() {
char a[80];
 char b[80]; int i, j;
  cout << "Enter a sentence: ";

gets(a); // Read a token at a time from the string.



for(i=0;; i++) { /* Read characters until either a space or the null
terminator is encountered. */

for(j=0; a[i]!=' ' && a[i]; j++, i++)

 b[j] = a[i];
```

```
 b[j] = '\0'; // null terminate the token

      b[j] = '\0'; // null terminate the token

     cout << b << '\n';

 if(!a[i])
  break;

 }
getch();

  return 0;
}
```

Solution:

We should use some **primitive data types in any language to make it more orthogonal** the above code is not orthogonal because the same functionality could be achieved by using pointers. It means that a relatively small set of primitive constructs can be combined in a relatively small number of ways and they are independent of the context of their appearance in a program. Following changes in the code can make it more orthogonal w.r.t simplicity and efficiency.

```
#include <iostream.h>

#include <stdio.h>

# include <conio.h>

 using namespace std;



int main() {

char a[80];

 char b[80];

  char *p, *q;

 int i, j;

  cout << "Enter a sentence: ";
```

```
gets(a); // Read a token at a time from the string.

p=a;

 // Read a token at a time from the string.


while( *p ) {


q = a ; // set q pointing to start of token

/* Read characters until either a space or the  null terminator is encountered */


while ( *p != ' ' && *p ) {


      *q = *p;


q++; p++;


   }


if( *p ) p++; // advance past the space



 *q = '\0'; // null terminate the token
```

```
cout << a<< '\n';

}


getch();

 return 0;


}
```

The End