
Problem 1: What do you understand by imperative programming programming languages? How is it related to von Neumann architecture? Draw a schematic of von Neumann architecture and explain.

Problem 2: For *C-programming*, discuss the following with an example of yours: *If-statement* is not mandatory in *C-programming*, given that *while* control statement is available.

Problem 3: Discuss the following criteria of efficient language design criteria with example of each. Select a programming language of your preference and evaluate it according to these criteria.

- Generality
- Uniformity
- Extensibility
- Restrictability

Problem 4: Discuss (in brief) the types of errors that generally occur in a program. Provide an example pseudocode for each type of error.

Problem 5: Write short notes on the followings:

(a) Why are compilers separated into front-end and back-end?

(b) von Neumann bottleneck

(c) What roles do symbol table have in compilers?

(d) Portability of programming languages.

Problem 6: What scoping strategies would you prefer while designing a programming language? Consider the below *C-program* and explain the printed number sequence using the concept of scoping. Assume that the syntax used in the program compiles and execute withouth any errors.

```
#include <stdio.h>
            int i = 6;
            int main(int argc, char * argv[])
 {
            printf("%d\n", i);
     {
            int i = 7, j = 8;
            printf("%d\n%d\n", i, j);
       {
            int i = 5;
            printf("%d\n%d\n", i, j);
         }
            printf("%d\n", i);
       }
                  printf("%d\n", i);
                  return 0;
  }
```

Problem 7: Write short description of each of the below programming language. Your MUST include the advantages they offered during the time of their initiation. Also, note down the shortcomings/limitations (if there's any).

- Short-code
- Assembly Language
- Fortran
- ALGOL 58-60
- C/C ++
- Java
- Python
- LISP/Scheme