2019-02 C Programming (CSE2035)

Project 2 : Interpreter Development2

Saejoon Kim, PH.D.

Dept. of Computer Science and Engineering

Sogang University

1. Design goal

The interpreter is a computer program or environment that directly executes the source code of a programming language. Through this project, we will develop an interpreter that performs simple operations and practice the dynamic allocation, pointer, and string processing that we learned in class.
In this project, students create their own interpreters with simple operations. The program has the following restrictions.

- Development Environment : Linux (cspro server)
- Use pointer and dynamic allocation to solve a given problem.

The production of this program should follow the basic course of the C programming class and the results should be submitted in accordance with the published format (source files and documents).

2. Requirements

2.1 Preparation

Define what needs to be implemented in this program and investigate the grammatical knowledge of C language required for production. Learn about the environment and constraints in which production is made, how this affects program development, and what additional features are available.

2.2 Analysis

Design a function to meet the program's requirements, after the definition of the problem to be implemented and the acquisition of basic knowledge has been completed. To modularize the problem, the function, parameters, return values, etc. of the function are accurately identified and considered how each function is implemented.

2.3 Development

Once the analysis is over, it goes into production. Implement each function to perform the function defined in the analysis step. If problems occur that were not found beforehand during production, they can be solved flexibly.

/* Explain in detail what function you have implemented and how it works */
/* If you don't explain it, we'll assume you didn't implement it */
NODE *head,*tail,*pNew,*pPre is global variable.
and NODE includes name[],value[],value_number ,*link.

int main()
- similar to assignment1
character array op1 is statically allocated and it stores command with the keyboard.

Using the while loop, the program repeated this.

1.The program reads the command with keyboard, and stored it to op1.
2.. If op1 is "LIST", LIST();
3. If op1 is "PRINT",PRINT();
4. If op1 is "INSERT",INSERT();
5. If op1 is "SORT",SORT();
6. If op1 is "EXIT", break the loop.
7.else, print "error₩n".
after breaking the loop, the program ends.

LIST()
- this function stores integer values.

-the integer values are stored using linked list(NODE). if head is NULL(the initialized value of head is NULL), dynamically allocated it using malloc() included at <stdlib.h> . and pPre is head , head->link=tail.
-the program stores input to 'op' and the first part of op is op1 (using sscanf())
- check if op1 is "LIST","PRINT","INSERT","SORT","EXIT" because the name of command can't be used as variable name. and if it is, print "error₩n" and return.
-if it isn't, op2 is the last part of op. and op2[strlen(op2)-1]='₩0'
-Using 'while loop'and atoi(), stored the integer values at op2 to pNew->value
-and the number of input can't be more than 32, so if it is(using parameter'j'), print "error₩n" and return.
-if it isn't, pNew->value_number=j, and link pNew and pPre and tail (pPre->link=pNew, pNew->link=tail)
and pPre=pPre->link.
-The program releases op,op2 and return.

PRINT()
-the function prints a stored variable.

varname[] is variable name and the program stores input to varname.
the initialized value of target is head and using while loop, the program checks if target->value_number is varname(target :from head to tail)using strcmp included at <string.h>
if there isn't then print"not found₩n" and return.
if there is, print target->value using for loop and target->value_number and return .
if target->value_number is 0, print "empty₩n" and return.

INSERT()
-This command inserts integers value into the list at the specified index.

the program stores input to varname.
the initialized value of target is head and using while loop, the program checks if target->value_number is varname(target :from head to tail)using strcmp included at <string.h>
if there isn't then print"not found₩n" and return.
if there is, stores index and check if the index value is 0~list length.
if it isn't, print "error₩n" and return. if it is, the program stores index_value with the keyboard using fgets included at<string.h>.
and stores integer values to plus_value using atoi() and plus_number and while loop.
(if index_value[i-1]==' '&&index_value[i]!=' ', plus_value[plus_number] = atoi(index_value + i); plus_number++;)
and if plus_number+target->value_number>32), print "error₩n" and return.
else, insert plus_value to target->value and target=target->link and return.

SORT()
-This function sorts all elements in the list.

varname[] is variable name and the program stores input to varname.

the initialized value of target is head and using while loop, the program checks if target->value_number is varname(target :from head to tail)using strcmp included at <string.h>
if there isn't then print"not found₩n" and return.
if there is, the program stores value to flag and check if flag is 1 or -1. if it isn't, print "error₩n" and return.
if it is, using bubble sort, if flag is 1, the program sorts list to ascending order.
if flag is -1, the program sorts list to descending order.

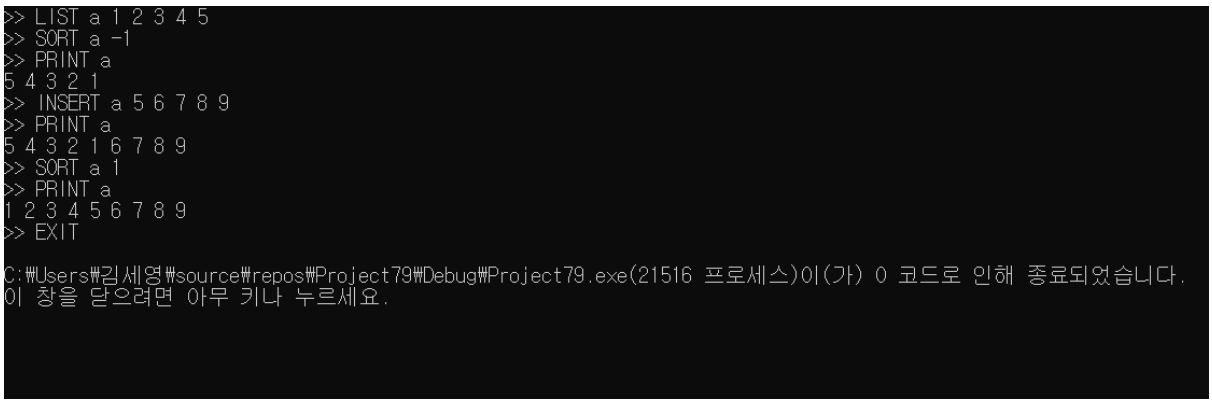
2.4 Test

Whether the program produced is actually working and running according to the rules presented is the biggest criterion for evaluation. It is also considered whether all other possible exceptions have been handled, whether the problem has been correctly identified at the design stage, and whether the solutions have been reasonable and clear.

2.5 Evaluation

Enter several test cases and evaluate them.

/* Execute your program and show the results of your test cases */
/* and explain it with screenshots. */



1. a command is "LIST" and the function stores integer value 1,2,3,4,5
2. flag is -1, so a is listed to descending order.
3. a command is "PRINT" so the program prints 5 4 3 2 1 (descending order)

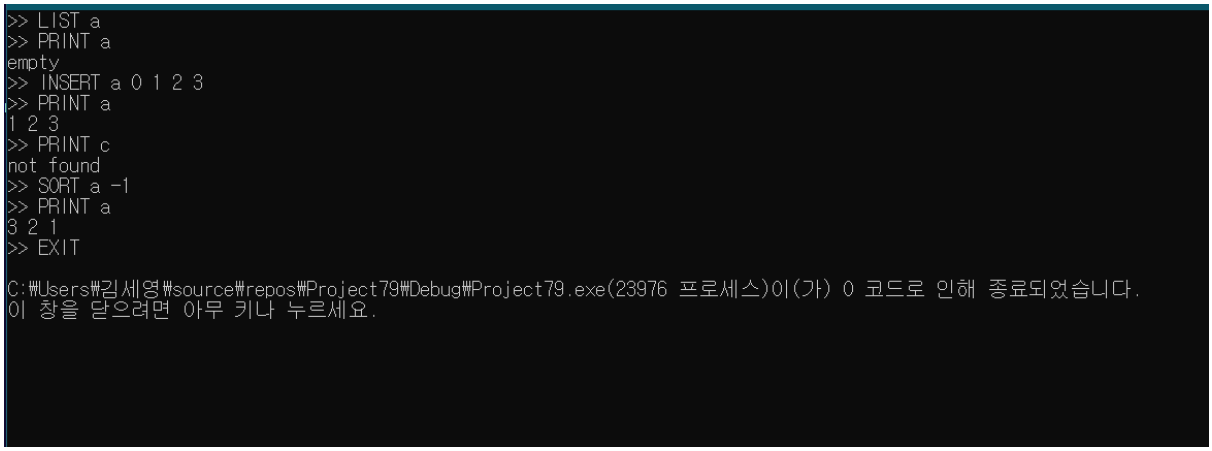
4.a command is insert and insert index is 5, and integer values are 6,7,8,9. so list 'a' is 5 4 3 2 1 6 7 8 9.

5. a comand is PRINT, so the program prints 5 4 3 2 1 6 7 8 9.

6. flag is 1, so a is listed to ascending order.

7.a comand is PRINT, so the program prints 1 2 3 4 5 6 7 8 9.

8.a command is "EXIT" so the program ends.



1. a is empty list.

2. there is no element is the list, so print "empty₩n"

3. a command is "INSERT" and index is 0, so a is 1 2 3.

4. a command is "PRINT", so print 1 2 3.

5.c is not found, so print "not found₩n".

6. flag is -1,so a is listed descending order.

7. a command is PRINT so print 3 2 1

8.a command is "EXIT", so program ends.

2.6 Stability

It is required to identify various constraints and devise programmatic treatments that can complement situations where errors can be caused (C-language buffer overflow vulnerabilities). For this, students should create separate functions. After the completion of the actual program, the functions shall be possible according to the project details in the test process, and the modifications and completeness of the parts not properly performed shall be continuously carried out.

2.7 Durability

In carrying out the project, students can accumulate basic knowledge of C language through learning about the pointer, dynamic arrangement, and string processing of C language and learn the principle of the operation of the interpreter in the process of coding themselves. Students can also actively handle the various errors that occur during the project and correct them on themselves.

2.8 Standard

Basically, the project is built in compliance with the ANSI C standard of the gcc Compiler(5.4), and students connect to the Linux server and proceed with the project, so they provide a desktop and ssh interface program that can access that server. Sub-accounts are issued to each student on the Linux server to provide an environment where they can freely use their allocated capacity to proceed with the project.

3. Note

3.1 Environment

Students connect to a Linux server and proceed with the project, so they provide a desktop and ssh access program to connect to that server. Sub-accounts are issued to each student on the Linux server that they connect to to provide an environment where they can freely proceed with the project based on the capacity they receive.

3.2 Focus

As a programing class that follows the first semester, the focus should be on generating interest from the sense of achievement in the completion of the project, including an additional simple algorithm in the grammatical part of programming, and improving the ability to complete fragmentary knowledge that has been learned during the two semesters in a single project.

3.3 Team

Individuals form a team.

3.4 Due-date

Sun. December 8, 23:59:59
You must be able to complete the program and write a report within the stated project period.