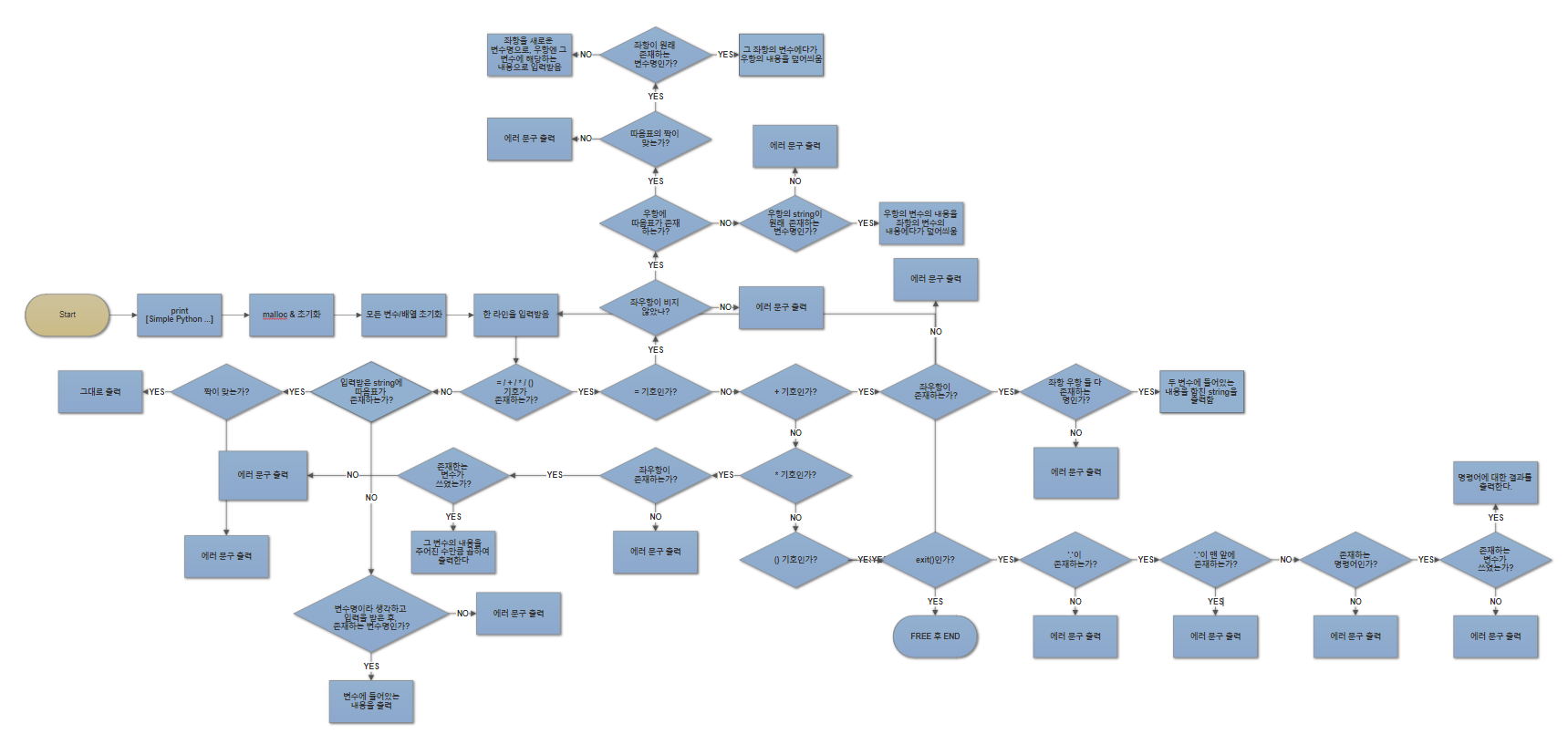
1. 프로그램 실행 흐름도 (Flowchart)

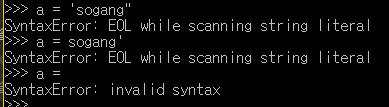


1. 프로그램의 설명

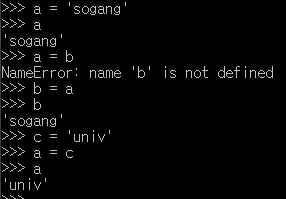
이번 프로젝트는 C언어로 Python Interpreter를 만드는 것이 목적이었다. 그 때문에, 작성한 프로그램이 python method들에 대하여 정확하게 동작하는 것이 제일 중요한 기준이 되었다. 앞으로 이러한 인터프리터가 어떻게 작용하는지를 설명하겠다.

첫 단계로는 "[Simple Python Interpreter implemented by C]\n" 라고 출력을 한 다음, 변수 설정과 필요한 string들을 위한 malloc과 변수들의 초기화를 해주었다. 변수는 총 20개를 받을 수 있었으므로, typedef struct로 만든 string이라는 구조체를 변수의 이름과 내용을 엮은 구조체로 만든 다음 그러한 구조체를 20개를 배열로 만들었다. (malloc을 이용하여 메모리를 할당해 주었다.) 그런 다음 while문의 무한루프를 돌려 exit()가 나올 때까지 루프가 끝나지 않도록 하였다. 또한, 루프가 시작될 때 마다 ‘>>> ’를 출력하여 주어 python과 같아 보이도록 하였다. 그렇게 한 줄을 입력 받고, 그 입력 받은 string에 =/+/\*/()의 기호가 존재하는지 확인하여 보았다.

= 가 포함 되어있는 string을 입력 받았을 경우, =기호 우항에 따옴표(크거나 작은)가 존재하는지를 확인하여 보았다. 만약 있다면, 짝이 맞나 맞지 않나를 따져 맞지 않으면 "SyntaxError: EOL while scanning string literal\n" 를 출력하여 오류가 발생하였음을 알렸다. 만약 짝이 맞았다면, =기호의 좌항에 있는 ‘단어’가 원래 존재하는 변수의 이름인지를 확인하여 봤다. 만약 맞다면, 좌항의 변수명에 해당하였던 변수의 내용을 우항의 따옴표 안에 있던 내용으로 덮어씌웠다. 만약 그렇지 않았다면, 좌항을 새로운 변수의 변수명으로 설정하고, 우항의 따옴표 안에 있던 내용을 새로운 변수의 내용으로 집어넣었다. 그런 다음, 다음 배열칸으로 넘어가도록 i++를 해주었다. (i는 배열의 주소를 가리키는 숫자로 설정하였다.) 만약 =기호 우항에 따옴표가 존재하지 않았다면, 좌우항 모두 변수명을 가리키는 것으로 인식하였다. 그러나 만약 우항에 아무것도 없이 ‘a = ‘만의 형태인 string이 입력 받았을 땐, 다음과 같은 에러 문구를 출력하게 하였다. "SyntaxError: invalid syntax\n" 만약 NULL값이 아니라면, 우항의 변수명의 존재유무를 따졌다. 만약 우항의 변수명이 존재하지 않는 변수명이라면, 다음과 같은 에러 문구를 출력하게끔 하였다. "NameError: name '%s' is not defined\n" 에서 %s에는 우항의 변수명이 들어가도록 하였다. 반면에, 만약 우항의 변수명이 존재하는 변수명이라면, 좌항의 변수명이 존재하는 변수명인가를 확인 하여 보았다. 만약 원래 존재하는 변수명이라면, 좌항의 변수의 내용에다가 우항의 변수의 내용을 덮어씌우게끔 하였다. 그러나 만약 원래 존재하지 않는 변수명이라면, 좌항의 변수명을 가진 새로운 변수를 만들어 그 변수의 내용에 우항의 변수명에 해당하는 변수의 내용을 넣게끔 하였다.



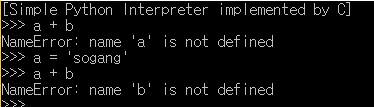
(그림1. 에러문구출력 경우)



(그림2. 에러 문구 출력 경우 및 덮어씌우기/새로 쓰기)

* 처음엔 a에 sogang이라는 내용이 들어갔었다. 그러나 a = b를 하자 b가 존재하지 않았기에 에러 문구가 출력이 되었다. 그러나 반대로 b = a를 하니 a에 있던 내용이 b에 씌워진 것을 알 수 있다. 또한, c라는 변수를 새로 만들고, a = c를 하였더니 a에 원래 있던 내용 대신 c의 내용이 덮어씌워진 것을 알 수 있다.

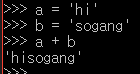
+ 가 포함 되어있는 string을 입력 받았을 경우, +기호의 좌우항이 다 존재하는 변수명인지를 확인해보았다. 만약 좌항만이 존재하지 않는다거나, 좌우항 둘 다 존재하지 않는다면 "NameError: name '%s' is not defined\n" 에서 %s에는 좌항의 ‘변수명’이 들어가도록 하였다. 만약 우항에 해당하는 변수명이 존재하지 않을 경우 "NameError: name '%s' is not defined\n" 에서 %s에는 우항의 ‘변수명’이 들어가도록 하였다. 만약 좌우항 둘 다 존재하는 변수명들일 경우, 좌항에 해당하는 변수의 내용과 우항에 해당하는 변수의 내용을 이은 string을 출력하도록 하였다. 또한, 만약 우항이 아예 없는 경우 "SyntaxError: invalid syntax\n" 라는 에러 문구를 출력하였다.



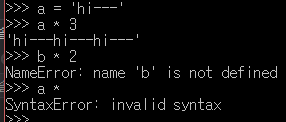
(그림3. 좌우항 둘 다 존재하지 않을 때 / 우항만이 존재하지 않을 때 에러문구출력)



(그림4. 우항이 없을 때 에러문구출력)



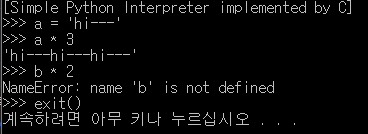
(그림5. 좌우항 둘 다 존재하는 경우 출력결과)

\*가 포함 되어있는 string을 입력 받았을 경우, \*기호의 좌우항 중 하나가 존재하는 변수명인지를 확인하였다. 만약 맞다면, 좌우항 중 숫자인 항을 참고하여 그에 해당하는 변수의 내용을 그 숫자만큼 곱한 결과를 출력하도록 하였다. 그러나 만약 아니라면, "NameError: name '%s' is not defined\n" 에서 %s에는 좌우항 중 숫자가 아닌 변수명을 나타내는 항으로 하였다. 

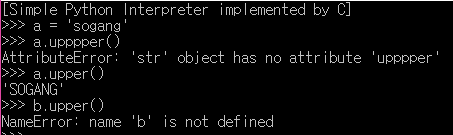
(그림6. 성공했을 때의 출력 경우 및 에러 출력 경우)

* a라는 변수에는 hi---라는 내용이 들어가 있지만, b라는 변수는 애초에 생성되지 않았기 때문에 에러 문구가 출력이 된다.

()가 포함 되어있는 string을 입력 받았을 경우, 일단 그 string이 exit()인지부터 파악하여 보았다. 만약 exit()라는 string이었다면 바로 while문의 무한루프에서 break하게끔 만들었다. 그렇게 무한루프에서 나오게 되면, 이전에 사용하였던 모든 malloc으로 할당한 메모리들을 free시켜준 다음, 프로그램을 종료 시키게끔 하였다. (특히, print라는 char\*변수는 malloc을 하지 않았다. 그 이유는 char\*값으로 리턴되는 함수들을 받는 변수였기 때문이다. 그런데, 이 변수는 프로그램이 실행된 후 끝까지 안 쓰일 때가 있어서 printcheck라는 변수를 사용하여 print라는 char\* 변수가 이용되었을 때에만 free시켜주는 것으로 하였다. 만약 print가 한번도 쓰이지 않았을 경우 free(print)를 해버리게 되면, malloc도 되지 않은 free할 것이 없는 포인터를 free시켜주는 것이기 때문이다.) exit()이 아니라면, 변수명.명령어()의 형태로 되어있음을 참고하여, 만약 명령어가 존재하지 않는 (짜라고 주어진 명령어 이외의 명령어이거나 오타) 상황에서는 에러 문구를 python에서처럼 출력하였다. "AttributeError: 'str' object has no attribute '%s'\n" 여기서 %s에는 존재하지 않는 명령어로 인식된 ‘명령어’가 들어가게끔 하였다. 만약 존재하는 명령어라면, 변수명 역시 존재하는가를 확인해 보았다. 만약 존재하는 변수명이면, 그 변수명을 이용한 그 명령어의 결과를 출력하도록 하였다. 그러나, 존재하지 않는 변수명이라 판단이 되면 다음과 같은 에러 문구를 출력하도록 하였다. "NameError: name '%s' is not defined\n" 에서 %s에는 존재하지 않는 변수명이 들어가게끔 하였다.



(그림7. exit()했을 때 바로 종료되는 경우)

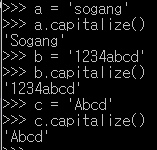


(그림8. 존재하지 않는 함수가 있을 경우 에러문구 및 변수의 존재여부에 따른 결과출력

* a라는 변수에는 이미 sogang이라는 내용이 들어가 있지만, b라는 변수는 애초에 생성이 되지 않았기 때문에 에러 문구가 출력이 된다. 또한 upppper라는 함수는 없기에

이번 프로젝트에서 사용된 python의 명령어의 역할을 나열하자면 다음과 같다.

capitalize(): string의 맨 앞 문자가 영어 소문자라면 영어 대문자로 바꾸어 준다.



(그림9. capitalize()의 역할)

swapcase(): string중 소문자를 대문자로, 대문자를 소문자로 바꾸어 준다.



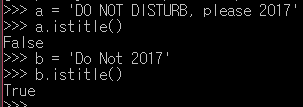
(그림10. swapcase()의 역할)

title(): 한 단어마다 첫 글자가 소문자라면 대문자로 바꾸어 준다. 그러나, python에서는 공백이 아닌 그저 알파벳이 아닌 문자라면 보편적인 띄어쓰기, 즉 단어의 구별을 해주는 것으로 보아 알파벳이 아닌 문자 뒤에 오는 알파벳도 대문자로 바꾸어 준다.



(그림11. title()의 역할)

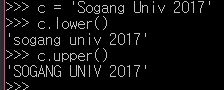
istitle(): 위에 언급한 title()의 결과값처럼 되어있는지, 즉 알파벳이 아닌 문자 뒤의 글자가 알파벳이라면 대문자인지 확인하여 맞으면 True, 아니면 False를 출력한다.



(그림12. istitle()의 역할)

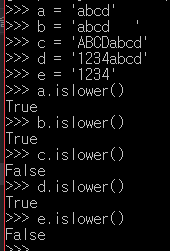
lower(): string중 대문자가 존재하면 그 문자를 소문자로 바꾸어 준다.

upper(): string중 소문자가 존재하면 그 문자를 대문자로 바꾸어 준다.



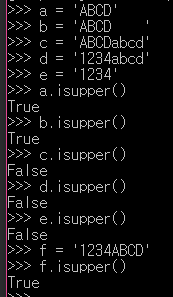
(그림13. lower() & upper()의 역할)

islower(): string중 알파벳이 있다면 소문자인지를 확인한다. 만약 알파벳이 존재하고 그 알파벳들 중 소문자밖에 없으면 True를 출력하고, 알파벳이 아예 string에 존재하지 않거나 알파벳 중 대문자가 존재하면 False를 출력한다.



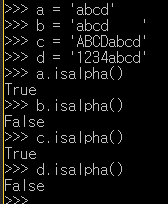
(그림14. islower()의 역할)

isupper(): string중 알파벳이 있다면 대문자인지를 확인한다. 만약 알파벳이 존재하고 그 알파벳들 중 대문자밖에 없으면 True를 출력하고, 알파벳이 아예 string에 존재하지 않거나 알파벳 중 소문자가 존재하면 False를 출력한다.



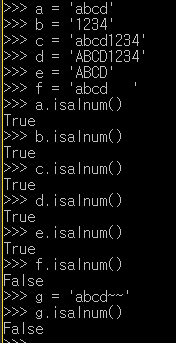
(그림15. isupper()의 역할)

isalpha(): string에 알파벳밖에 없는지를 확인한다. 만약 알파벳 외의 문자, 즉 숫자나 공백 같은 문자가 string에 존재하면 False, 순전히 알파벳밖에 없으면 True를 출력한다.



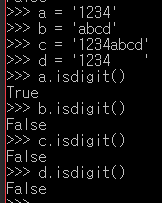
(그림16. isalpha()의 역할)

isalnum(): string에 알파벳이나 숫자밖에 없는지를 확인한다. 만약 알파벳이나 숫자 외의 문자, 즉 공백이나 특수문자 같은 문자가 string에 존재하면 False, 순전히 알파벳이나 숫자밖에 없으면 True를 출력한다.



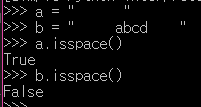
(그림17. isalnum()의 역할)

isdigit(): string에 숫자밖에 없는지를 확인한다. 만약 숫자 외의 문자, 즉 공백이나 알파벳 같은 문자가 string에 존재하면 False, 순전히 숫자밖에 없으면 True를 출력한다.



(그림18. isdigit()의 역할)

isspace(): string에 공백밖에 없는지를 확인한다. 만약 공백 외의 문자, 즉 숫자나 알파벳 같은 문자가 string에 존재하면 False, 순전히 공백밖에 없으면 True를 출력한다.



(그림19. isspace()의 역할)

lstrip(): string의 왼쪽 부분에 공백들이 있으면 공백 외의 문자가 나올 때까지 그 왼쪽의 공백들은 다 지운 다음 출력한다. 만약 string에 공백밖에 없을 시엔 ‘’만을 출력한다.



(그림20. lstrip()의 역할)

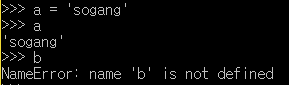
rstrip(): string의 오른쪽 부분에 공백이 있으면 오른쪽 부분의 공백들을 지운 후 출력한다. 만약 string에 공백밖에 없을 시엔 ‘’만을 출력한다.



(그림21. rstrip()의 역할)

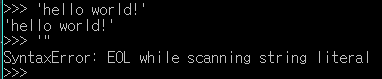
모든 명령어의 함수가 실행되고 나서는, 그 함수 안에서 쓰였던 할당된 메모리들을 free시켜주었다.

만약 입력 받은 string에 =/+/\*/()의 기호가 없다면, string에 따옴표(큰 따옴표나 작은 따옴표)가 존재하는지 확인하였다. 만약 존재하지 않는다면, string은 곧 변수명을 가리키는 것이라 인식하여 그 변수명이 존재하는 변수명인지를 확인하여 보았다. 만약 존재하는 변수명이라면, 그 변수의 내용을 출력하게끔 하였다. 그러나, 존재하지 않는 변수명일 시엔 에러 문구를 다음과 같이 출력하였다. "NameError: name '%s' is not defined\n" 에서 %s에는 존재하지 않는 변수명이 들어가게끔 하였다. 만약 따옴표가 존재한다면, 그 따옴표의 짝이 맞는지를 확인하여 보았다. 만약 짝이 맞지 않다면, 역시 위에 언급된 에러 문구를 출력한다. 그 문구는 다음과 같다. "SyntaxError: EOL while scanning string literal\n" 만약 짝이 맞다면, 입력 받은 string을 그대로 출력한다.



(그림22. 변수에 해당하는 값 출력 및 에러문구)

* a라는 변수에는 이미 sogang이라는 내용이 들어가 있어 a만이 입력이 되었을 때 그 내용이 출력이 되었다. 그러나, b라는 변수는 존재하지 않기 때문에 b만 입력이 되었을 때 에러 문구가 출력이 되었다.



(그림23. 따옴표가 있을 때 짝이 맞는 경우와 맞지 않는 경우의 결과 출력)