Indian Institute of Information Technology, Nagpur

Department of Computer Science and Engineering

Session:2019-2020
Course: Compiler
Lab Assignment 3

We were required to implement a semantic analyzer for C language. The test cases have been highlighted below:

Test case 1 (We cannot use the reserved keyword or identifier.)

Input

```
#include <stdio.h>
int main()
{
        float float=4;
        return 0;
}
```

Output

```
(base) abhibhas-MacBook:compiler_assgn abhibhagupta$ ./a.out rule1.c

Line 4 : syntax error float

Error : Undeclared Variable 4 : Line 4

Parsing Failed

Identifier              Type                    Parameter type(for functions)
main                     - INT
```

Corrected Program

Input

```
#include <stdio.h>
int main()
{
        float a=4;
        return 0;
}
```

Output

```
(base) abhibhas-MacBook:compiler_assgn abhibhagupta$ ./a.out -rule1.c

Parsing Completed

Identifier              Type                    Parameter type(for functions)
a                       FLOAT
main                     - INT
```

Test Case 2 (Arithmetic operations require numbers/integers.)

Input

```
#include <stdio.h>
int main()
{
        float a=4;
        a=2+'c';
        return 0;
}
```

Output

```
(base) abhibhas-MacBook:compiler_assgn abhibhagupta$ ./a.out rule2.c

Line 5 : syntax error '

Undeclared Variable c : Line 5

Parsing Failed

Identifier              Type                    Parameter type(for functions)
a                       FLOAT
main                     - INT
```

Corrected Program

Input

```
#include <stdio.h>
int main()
{
        float a=4;
        float b;
        b=2+a;
        return 0;
}
```

Output

```
[(base) abhibhas-MacBook:compiler_assgn abhibhagupta$ ./a.out -rule2.c

Parsing Completed

Identifier              Type                    Parameter type(for functions)
a                       FLOAT
b                       FLOAT
main                     - INT
```

Test Case 3 (New declarations don't conflict with earlier ones)

Input

```
#include <stdio.h>
int main()
{
        float a=4;
        float a=3;
        return 0;
}
```

Output

```
[(base) abhibhas-MacBook:compiler_assgn abhibhagupta$ ./a.out rule3.c

Error : Redeclaration of a : Line 5

Parsing Failed

Identifier              Type                    Parameter type(for functions)
a                       FLOAT
main                     - INT
```

Corrected Program

Input

```
#include <stdio.h>
int main()
{
        float a=4;
        float b=3;
        return 0;
}
```

Output

```
[(base) abhibhas-MacBook:compiler_assgn abhibhagupta$ ./a.out -rule3.c

Parsing Completed

Identifier              Type                    Parameter type(for functions)
a                       FLOAT
b                       FLOAT
main                     - INT
```

Test Case 4 (Break and Continue statements only appear in loops)

Input

```
#include <stdio.h>
int main()
{
        float a=4;
        if(a==2)
        {
                a=10;
        }
        else
        {
                a=7;
        }
        break;
        return 0;
}
```

Output

```
[(base) abhibhas-MacBook:compiler_assgn abhibhagupta$ ./a.out rule4.c

Undeclared Variable break : Line 13

Error : Undeclared Variable break : Line 13

Parsing Failed

Identifier              Type                    Parameter type(for functions)
a                       FLOAT
main                     - INT
```

Corrected program

Input

```
#include <stdio.h>
int main()
{
	float a=4;
	if(a==2)
	{
		a=10;
	}
	else
	{
		a=7;
	}

	return 0;
}
```

Output

```
(base) abhibhas-MacBook:compiler_assgn abhibhagupta$ ./a.out -rule4.c

Parsing Completed

Identifier              Type                    Parameter type(for functions)
a                       FLOAT
main                     - INT
```

Test Case 5 (The actual parameter's type in a call must be compatible with the formal parameter's type.)

Input

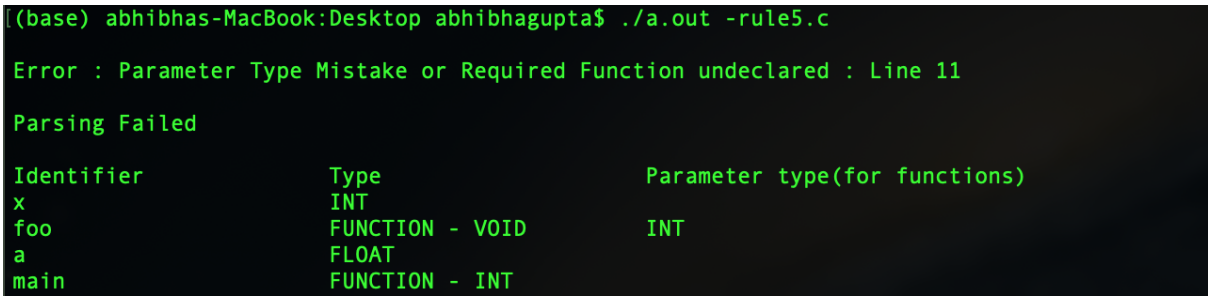
```
#include <stdio.h>

void foo(int x)
{
	return;
}

int main()
{
	float a=9;
```

```
    foo(a);
    return 0;
}
```

Output



Corrected Program

Input

```
#include <stdio.h>

void foo(float x)
{
    return;
}

int main()
{
    float a=9;
    foo(a);
    return 0;
}
```

Output