Compilers Project Proposal

Team 14

Names:

1. Radwa Samy	SEC:1 BN:18
2. Salma Ibrahim	SEC:1 BN:26
3. Mariam Hesham	SEC:2 BN:21
4. Nehal AbdAlkader	SEC:2 BN:28

Link of Video:

https://drive.google.com/file/d/1Yx4F8AJ2pmMZslHmWMewijb7y5en6aYN/view?usp=sharing

How to run:

Run the C# code if it contains a semantic error it should show an error in the gui, if it contains a syntax error just run in the hello.exe, and symbol table will be shown in a file called 'SymbolTable.txt' and the equivalent quadruples will be shown in the qui or in a file called 'out.txt'

Syntax Rules:

Our language should contain ';' at the end of the statement

1. **Variables declaration**
   - We have four types of variable (int, float, char, string)
   - Example declaration without initialization: int x;
   - Example declaration with initialization: int x = 9; / int x = y; / int x = y + 7;
2. **Constants declaration**
   - With word const followed by variable declaration
   - Example: const int x = 9;
3. **Mathematical and logical expressions**
   - We have math operations (+, -, /, *) and logical operations (&, |) can be used in assignment statement, parameters of function or in conditions.
   - example: 6 + x / 8 *6
4. **Assignment statement**
   - Starts with identifiers and equal sign we assumed identifier name can consist only from one char (capital or small) to minimize the symbol table and make it easy to control
   - Example: x = 6;/ x = 9+2;
5. **Conditional statement**
   - if (condition){statement} else if (condition){statement}/if (condition){statement} else {statement}/if (condition){statement}
   - while (condition){statement}
   - repeat-until loops : do { statement } while (condition);
   - for (statement; condition; statement) { statement }
   - switch (ID) {case values: statement break; default: statement}
6. **Functions**
   - Example:
     - Prototype: def get(int);
     - Function_call: get(5); or int x = get();

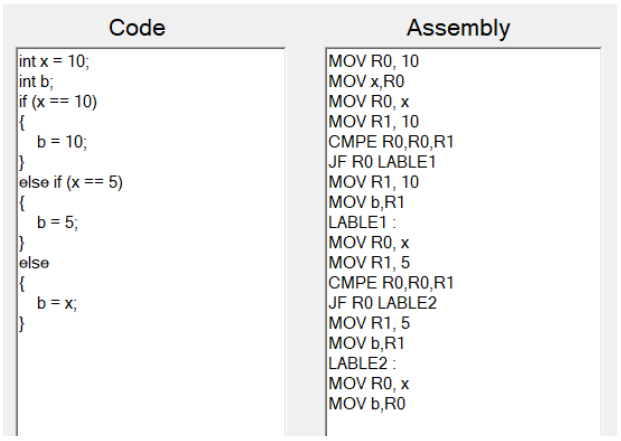
- Declaration function: def get(int x){return 7;}

Test Cases:

1. if.cpp

- when assigning variable, we take the value and put it in register then move that register to the variable position.
- ‘int b;’ no equivalent assembly for it, we just save that we declared variable with this name in symbol table.
- Each condition after CMPE we jump to the last label if this the register that contains the result of comparison is false with JF R0 LABLE1 (Jump if R0 is false to lable1)
- Symbol table:

Variable	Type	Scope	is_init	is_const
b	int	0	0	0
x	int	0	1	0



Code

```
int x = 10;
int b;
if (x == 10)
{
   b = 10;
}
else if (x == 5)
{
   b = 5;
}
else
{
   b = x;
}
```

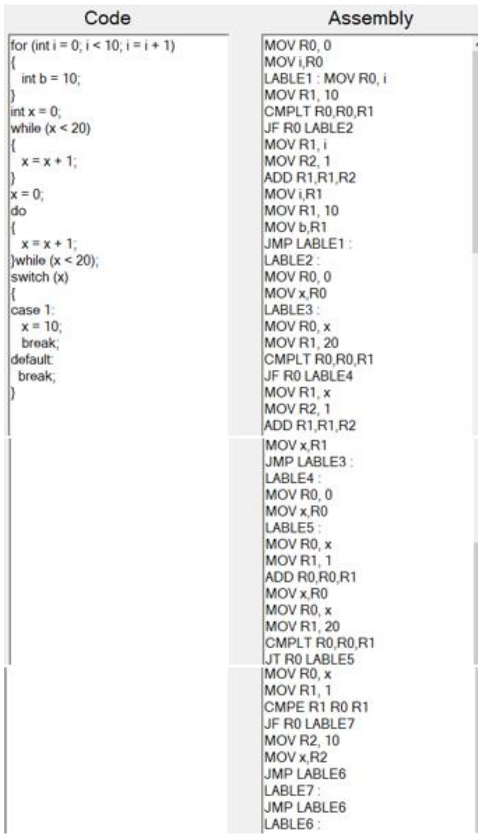
Assembly

```
MOV R0, 10
MOV x,R0
MOV R0, x
MOV R1, 10
CMPE R0,R0,R1
JF R0 LABLE1
MOV R1, 10
MOV b,R1
LABLE1 :
MOV R0, x
MOV R1, 5
CMPE R0,R0,R1
JF R0 LABLE2
MOV R1, 5
MOV b,R1
LABLE2 :
MOV R0, x
MOV b,R0
```

2. loops.cpp

- There are no errors except it finds out missing semi column and I and it.
- Symbol table:

Variable	Type	Scope	is_init	is_const
i	int	1	1	0
b	int	1	1	0
x	int	0	1	0



Code

```
for (int i = 0; i < 10; i = i + 1)
{
  int b = 10;
}
int x = 0;
while (x < 20)
{
  x = x + 1;
}
x = 0;
do
{
  x = x + 1;
}while (x < 20);
switch (x)
{
case 1:
  x = 10;
  break;
default:
 break;
}
```

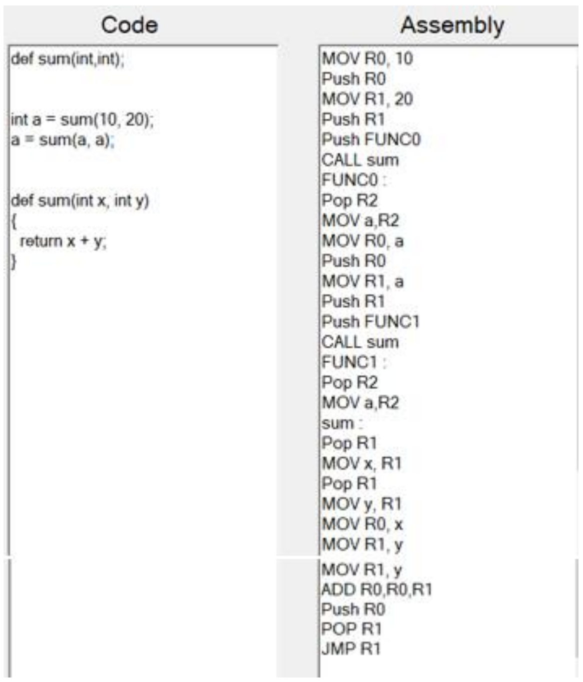
Assembly

```
MOV R0, 0
MOV i,R0
LABLE1 : MOV R0, i
MOV R1, 10
CMPLT R0,R0,R1
JF R0 LABLE2
MOV R1, i
MOV R2, 1
ADD R1,R1,R2
MOV i,R1
MOV R1, 10
MOV b,R1
JMP LABLE1 :
LABLE2 :
MOV R0, 0
MOV x,R0
LABLE3 :
MOV R0, x
MOV R1, 20
CMPLT R0,R0,R1
JF R0 LABLE4
MOV R1, x
MOV R2, 1
ADD R1,R1,R2
MOV x,R1
JMP LABLE3 :
LABLE4 :
MOV R0, 0
MOV x,R0
LABLE5 :
MOV R0, x
MOV R1, 1
ADD R0,R0,R1
MOV x,R0
MOV R0, x
MOV R1, 20
CMPLT R0,R0,R1
JT R0 LABLE5
MOV R0, x
MOV R1, 1
CMPE R1 R0 R1
JF R0 LABLE7
MOV R2, 10
MOV x,R2
JMP LABLE6
LABLE7 :
JMP LABLE6
LABLE6 :
```

3. functionss.cpp

- First we have to make a prototype at the top of the code main
- But declaration of the function should be at the bottom
- We define function with ‘def’ word before the name of the function
- When calling function, we push all parameter of the function in the queue (First in First out) and in the function declaration we pop all back
- Also we push the address of the returned line (Like PUSH FUNC0) and pop it before return
- Symbol table:

Variable	Type	Scope	is_init	is_const
a	int	0	1	0
x	int	1	0	0
y	int	1	0	0



Code

```
def sum(int,int);


int a = sum(10, 20);
a = sum(a, a);


def sum(int x, int y)
{
 return x + y;
}
```

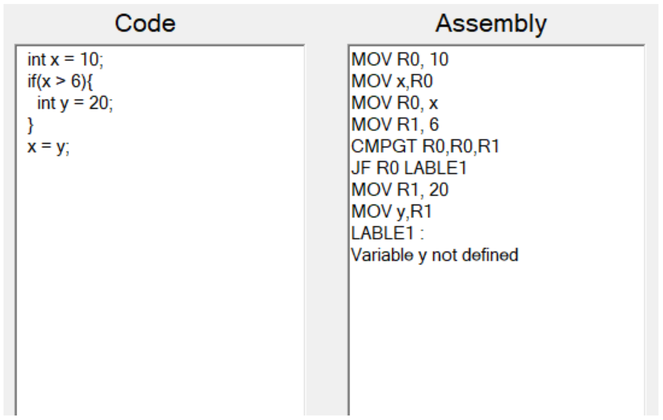
Assembly

```
MOV R0, 10
Push R0
MOV R1, 20
Push R1
Push FUNC0
CALL sum
FUNC0 :
Pop R2
MOV a,R2
MOV R0, a
Push R0
MOV R1, a
Push R1
Push FUNC1
CALL sum
FUNC1 :
Pop R2
MOV a,R2
sum :
Pop R1
MOV x, R1
Pop R1
MOV y, R1
MOV R0, x
MOV R1, y
MOV R1, y
ADD R0,R0,R1
Push R0
POP R1
JMP R1
```

4. blocks_error.cpp

- We didn’t understand block in document like that we thought blocks is equivalent to block of if and other condition so I write it in if block
- Here in this example the scope of y is so we can’t use it outside the block
- I didn’t know where to show error so I add it to assembly.
- Symbol table:

Variable	Type	Scope	is_init	is_const
x	int	0	1	0
y	int	1	1	0



Code

```
int x = 10;
if(x > 6){
  int y = 20;
}
x = y;
```

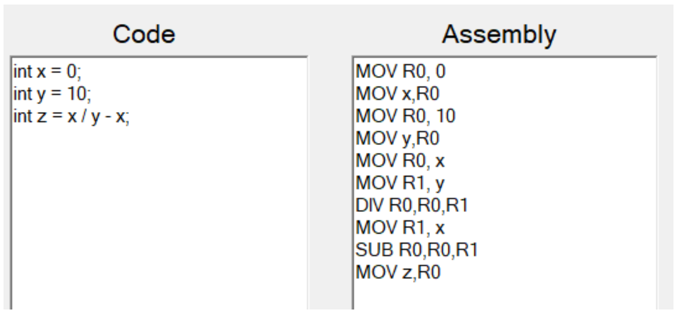
Assembly

```
MOV R0, 10
MOV x,R0
MOV R0, x
MOV R1, 6
CMPGT R0,R0,R1
JF R0 LABLE1
MOV R1, 20
MOV y,R1
LABLE1 :
Variable y not defined
```

5. Expression.cpp

- We didn’t implement the Boolean variable so we get error and not do any action it just stops at z
- Symbol table:

Variable	Type	Scope	is_init	is_const
x	int	0	1	0
y	int	0	1	0
z	int	0	1	0



Code

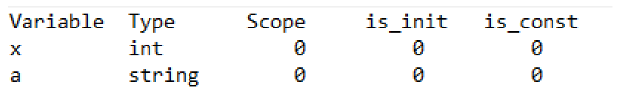
```
int x = 0;
int y = 10;
int z = x / y - x;
```

Assembly

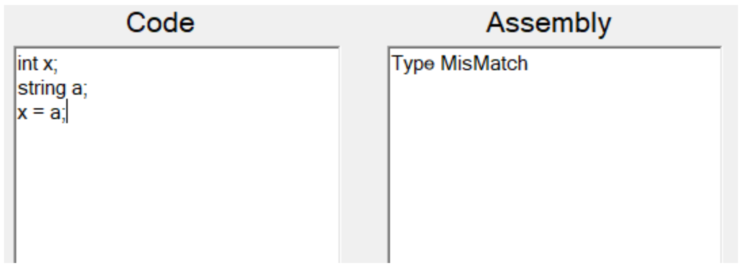
```
MOV R0, 0
MOV x,R0
MOV R0, 10
MOV y,R0
MOV R0, x
MOV R1, y
DIV R0,R0,R1
MOV R1, x
SUB R0,R0,R1
MOV z,R0
```

6. semantic_error.cpp

- We get type mismatch error as variable a and x are not from the same type.
- Symbol table:



```
Variable  Type      Scope     is_init   is_const
x         int          0         0         0
a         string       0         0         0
```

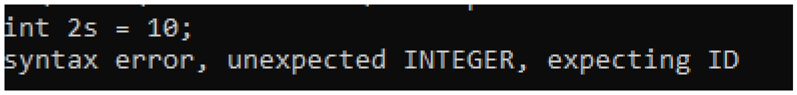


```
int x;
string a;
x = a;
```

```
Type MisMatch
```

7. synatx_error.cpp

- This case will show a syntax error as the variable name only contains letters
- Symbol table is empty



```
int 2s = 10;
syntax error, unexpected INTEGER, expecting ID
```

Quadruples:

Quadruple	**description**
MOV x,R1	Move value of R1 to variable x
JMP R0	Unconditional jump to the address in R0
JF R0 LABLE0	Jump to lable0 if R0 contains false
JT R0 LABLE0	Jump to lable0 if R0 contains True
PUSH R0	Push the value of R0 to the Queue
POP R0	Pop the value of the Queue and put it in R0
ADD,SUB,DIV,MUL R0,R1,R2	Do the math operation on the values in the registers R1,R2 and put the result in R0
CMPE R0,R1,R2	Compare the two values in R1 and R2 and but the comparing result in R0
CMPGT,CMPE,CMPGE,CMPLT,CMPLE	Compare(greater to >, equal ==, greater equal >= , less to <, less equal <=)
LABLE1: sum:	Labels that we have its address somewhere to jump to it (sum is label with the name of the function)

Task division:

Name	Task
Radwa Samy	Implement the math expressions and there priority of +/- & */÷ and also conditions (> < == && ||..etc)
Salma Ibraheem	Implementing function declaration and calling functions and create the gui
Mariam Hesham	Implementing conditions block like if ,while…..etc
Nehal AbdAllkader	Implementing the symbol table and assigning Variable to it and showing semantic errors