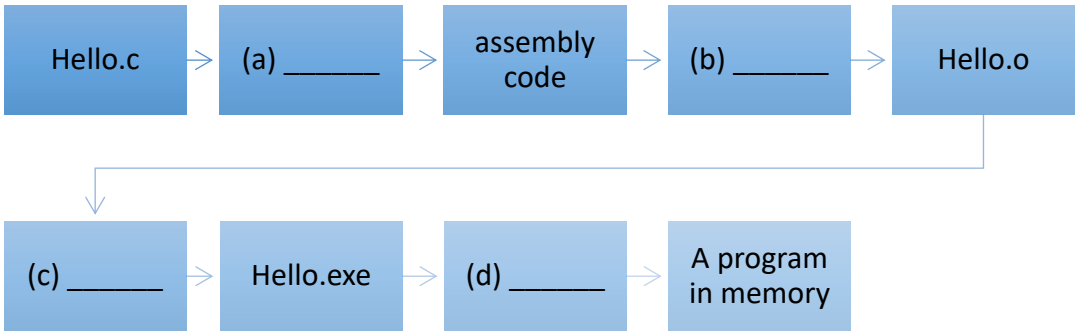
1. Fill the following blanks



2. Translate the statement "*a=b+3*" into assembly code. Suppose integer variable *a* is in memory space 0X00F0 and integer variable *b* is in memory space 0X00F8. And there are 8 registers, r0-r7, to use. The available assembly instructions are listed in the table below.

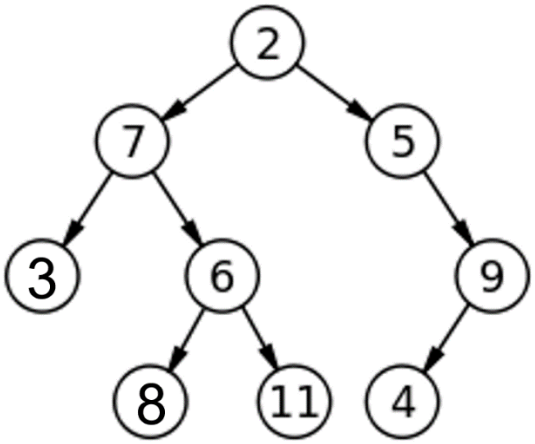
Opcode	Operand1	Operand2	Meaning
MOV	Register1	Register2	Move data from register2 to register1
	Register1	Constant	Set the value of register1 constant
	Register1	[Addr2]	Move the data (4 bytes) in memory addressed Addr2 to register1.
	[Addr1]	Register2	Move the data (4 bytes) from register2 to the memory addressed Addr1.
ADD	Register1	Register2	Add the values in register1 to register2 and store the result in register1

3. Explain how CPU, memory and registers work together to execute the statement "a=b+3". You may use the translated assembly code in question 2 to illustration the steps.

4. Given the pre-order and in-order traversal sequences of a binary tree:
   preorder:1 2 4 6 7 5 10 3 8 9
   inorder: 6 4 7 2 5 10 1 8 3 9
   show the structure of this binary tree.

5. Explain the purposes of the three steps "lexical analysis", "parsing process" and "code generation" in the compilation process.

6. Show the pre-order, in-order, and post-order traversal sequences of the following binary tree.



7. What is the time complexity of the following algorithm? Give your analysis

```
people = (int*)malloc(n*sizeof(int));
for(i=0;i<n;i++) people[i] = i+1;
remain = n;
current = 0;

while(remain>1){
    current = current + m-1;
    while (current>remain-1)
        current = current - remain;
    remain --;

    /* remove the killed node*/
    for (i = current; i< remain; i++)
       people[i] = people[i+1];
}
free(people);
```

8. Write the function BinarySearch using recursion. The data in array data is sorted ascendingly. If key is in data, return its index. Otherwise, return -1.

```
int BinarySearch(int data[], int start, int end, int key)
```