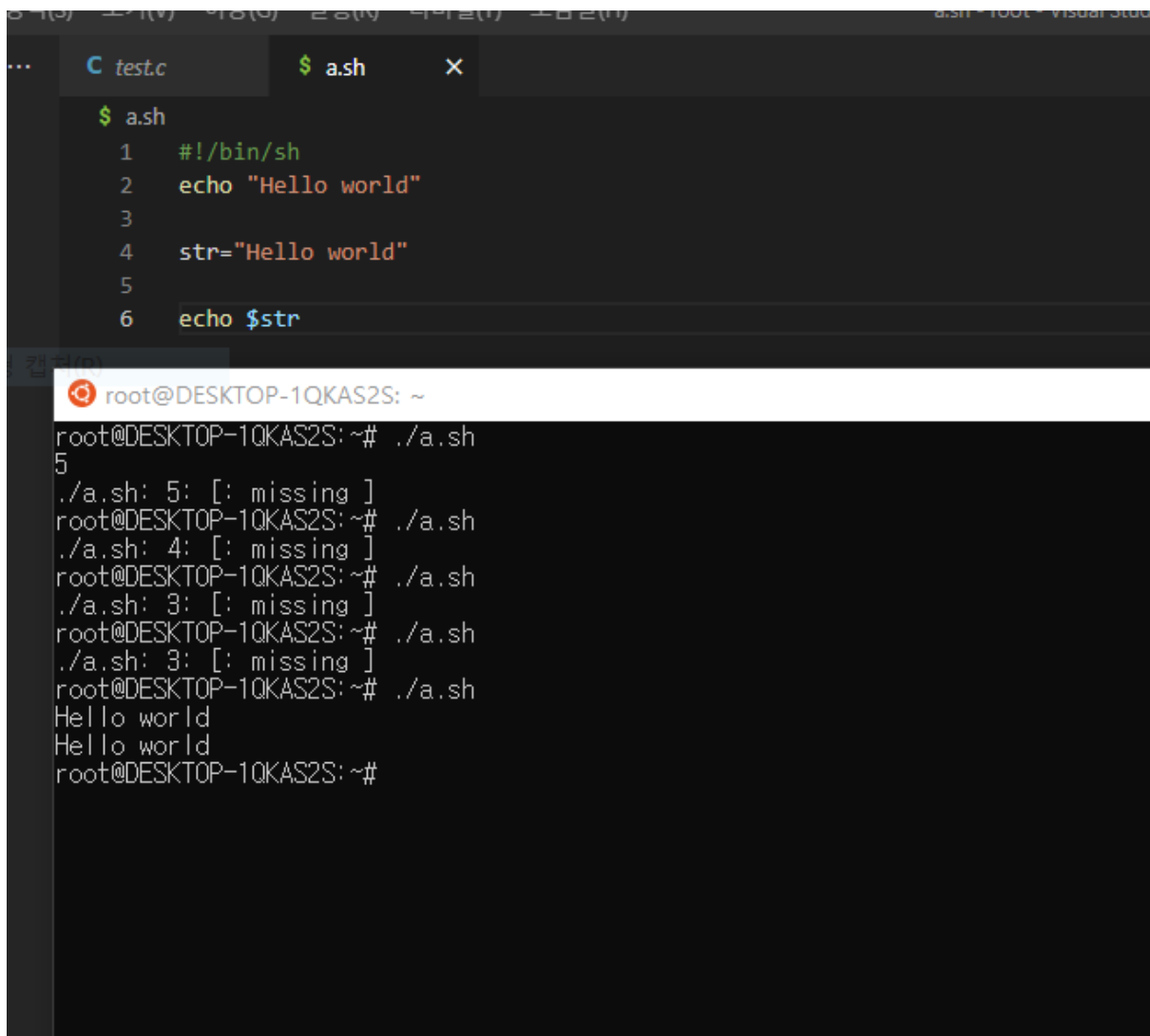


20213056_강민정 lab2

0번

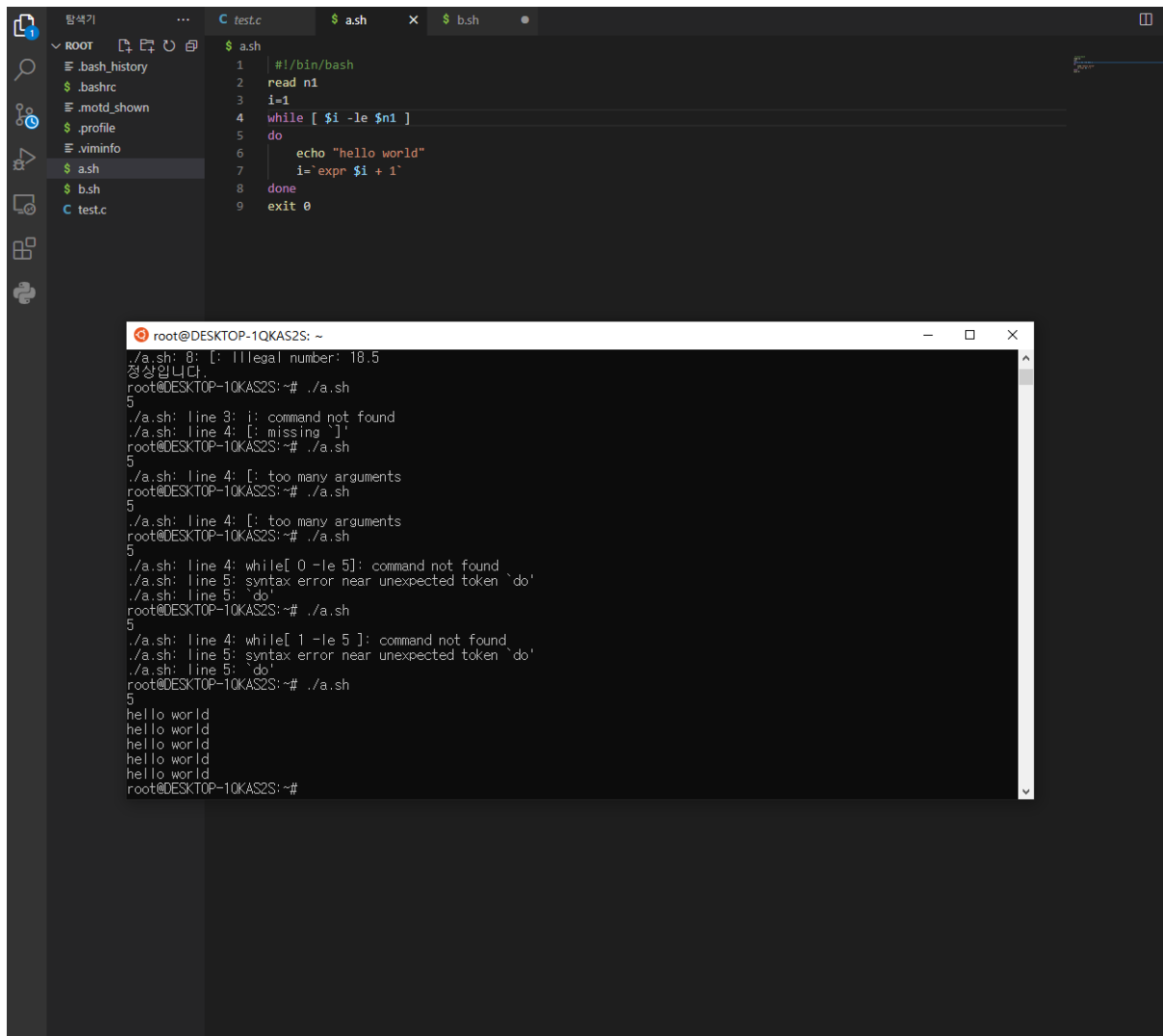


```
$ a.sh
1  #!/bin/sh
2  echo "Hello world"
3
4  str="Hello world"
5
6  echo $str

root@DESKTOP-1QKAS2S: ~
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: 5: [: missing ]
root@DESKTOP-1QKAS2S:~# ./a.sh
./a.sh: 4: [: missing ]
root@DESKTOP-1QKAS2S:~# ./a.sh
./a.sh: 3: [: missing ]
root@DESKTOP-1QKAS2S:~# ./a.sh
./a.sh: 3: [: missing ]
root@DESKTOP-1QKAS2S:~# ./a.sh
Hello world
Hello world
root@DESKTOP-1QKAS2S:~#
```

간단하게 출력문을 사용해서 Hello world를 출력했습니다. 별 어려움은 없었습니다.

1번



The screenshot shows a terminal window with a file explorer on the left. The file explorer lists files: `.bash_history`, `.bashrc`, `.motd_shown`, `.profile`, `.viminfo`, `a.sh`, `b.sh`, and `test.c`. The `a.sh` file is selected. The terminal window shows the following script content:

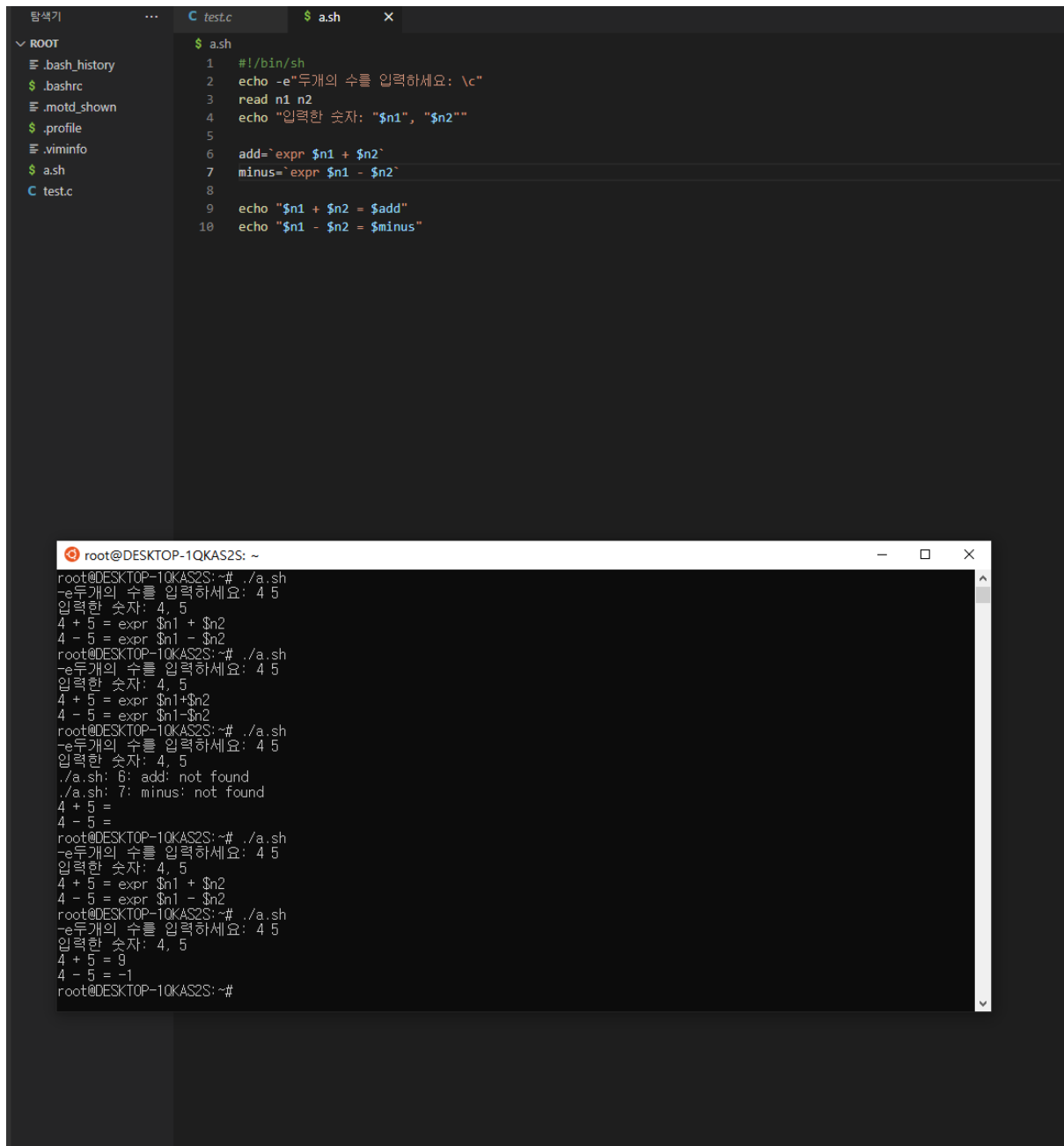
```
1 #!/bin/bash
2 read n1
3 i=1
4 while [ $i -le $n1 ]
5 do
6     echo "hello world"
7     i=`expr $i + 1`
8 done
9 exit 0
```

The terminal output shows the execution of the script with several errors and a successful loop execution:

```
root@DESKTOP-1QKAS2S: ~
./a.sh: 8: [: Illegal number: 18.5
정상입니다.
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: line 3: i: command not found
./a.sh: line 4: [: missing `]'
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: line 4: [: too many arguments
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: line 4: [: too many arguments
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: line 4: while[ 0 -le 5]: command not found
./a.sh: line 5: syntax error near unexpected token `do'
./a.sh: line 5: `do'
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: line 4: while[ 1 -le 5]: command not found
./a.sh: line 5: syntax error near unexpected token `do'
./a.sh: line 5: `do'
root@DESKTOP-1QKAS2S:~# ./a.sh
5
hello world
hello world
hello world
hello world
hello world
root@DESKTOP-1QKAS2S:~#
```

반복하기 위한 인자를 받아 그 인자를 기준으로 반복문을 돌려 출력했습니다.

2번

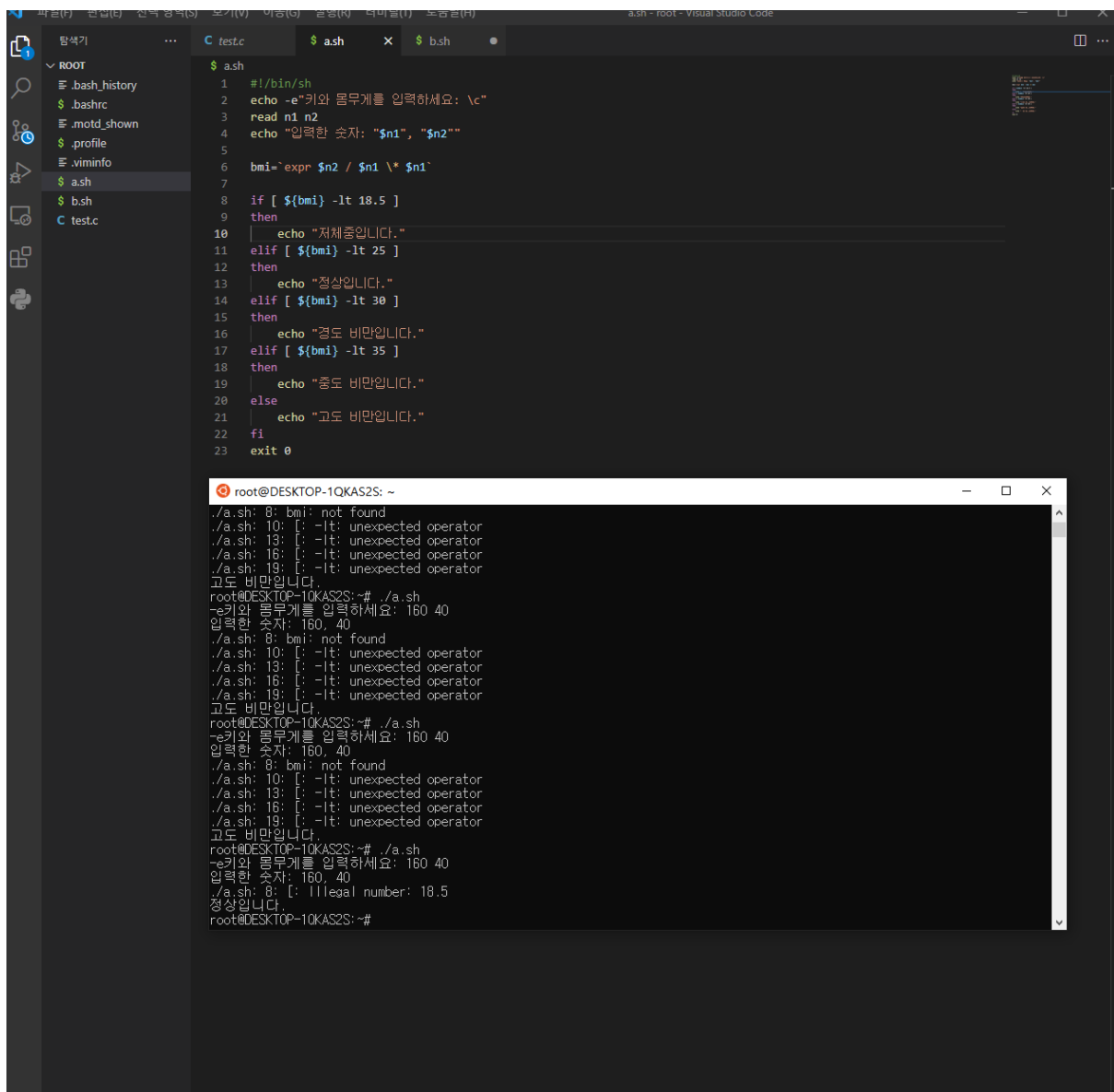


```
root@DESKTOP-1QKAS2S: ~  
root@DESKTOP-1QKAS2S: ~# ./a.sh  
-e두개의 수를 입력하세요: 4 5  
입력한 숫자: 4, 5  
4 + 5 = expr $n1 + $n2  
4 - 5 = expr $n1 - $n2  
root@DESKTOP-1QKAS2S: ~# ./a.sh  
-e두개의 수를 입력하세요: 4 5  
입력한 숫자: 4, 5  
4 + 5 = expr $n1+$n2  
4 - 5 = expr $n1-$n2  
root@DESKTOP-1QKAS2S: ~# ./a.sh  
-e두개의 수를 입력하세요: 4 5  
입력한 숫자: 4, 5  
./a.sh: 6: add: not found  
./a.sh: 7: minus: not found  
4 + 5 =  
4 - 5 =  
root@DESKTOP-1QKAS2S: ~# ./a.sh  
-e두개의 수를 입력하세요: 4 5  
입력한 숫자: 4, 5  
4 + 5 = expr $n1 + $n2  
4 - 5 = expr $n1 - $n2  
root@DESKTOP-1QKAS2S: ~# ./a.sh  
-e두개의 수를 입력하세요: 4 5  
입력한 숫자: 4, 5  
4 + 5 = 9  
4 - 5 = -1  
root@DESKTOP-1QKAS2S: ~#
```

변수값을 받고 해당변수값을 이용해서 add 와 minus 의 연산을 정했습니다.

이후 만들어진 연산자를 사용해 값을 출력했습니다.

3번



```
1  #!/bin/sh
2  echo -e"키와 몸무게를 입력하세요: \c"
3  read n1 n2
4  echo "입력한 숫자: \"$n1\", \"$n2\""
5
6  bmi=`expr $n2 / $n1 \* $n1`
7
8  if [ ${bmi} -lt 18.5 ]
9  then
10 | echo "저체중입니다."
11 | elif [ ${bmi} -lt 25 ]
12 | then
13 | echo "정상입니다."
14 | elif [ ${bmi} -lt 30 ]
15 | then
16 | echo "경도 비만입니다."
17 | elif [ ${bmi} -lt 35 ]
18 | then
19 | echo "중도 비만입니다."
20 | else
21 | echo "고도 비만입니다."
22 | fi
23 exit 0
```

```
root@DESKTOP-1QKAS2S: ~
./a.sh: 8: bmi: not found
./a.sh: 10: [: -lt: unexpected operator
./a.sh: 13: [: -lt: unexpected operator
./a.sh: 16: [: -lt: unexpected operator
./a.sh: 19: [: -lt: unexpected operator
고도 비만입니다.
root@DESKTOP-1QKAS2S:~# ./a.sh
-e키와 몸무게를 입력하세요: 160 40
입력한 숫자: 160, 40
./a.sh: 8: bmi: not found
./a.sh: 10: [: -lt: unexpected operator
./a.sh: 13: [: -lt: unexpected operator
./a.sh: 16: [: -lt: unexpected operator
./a.sh: 19: [: -lt: unexpected operator
고도 비만입니다.
root@DESKTOP-1QKAS2S:~# ./a.sh
-e키와 몸무게를 입력하세요: 160 40
입력한 숫자: 160, 40
./a.sh: 8: bmi: not found
./a.sh: 10: [: -lt: unexpected operator
./a.sh: 13: [: -lt: unexpected operator
./a.sh: 16: [: -lt: unexpected operator
./a.sh: 19: [: -lt: unexpected operator
고도 비만입니다.
root@DESKTOP-1QKAS2S:~# ./a.sh
-e키와 몸무게를 입력하세요: 160 40
입력한 숫자: 160, 40
./a.sh: 8: [: Illegal number: 18.5
정상입니다.
root@DESKTOP-1QKAS2S:~#
```

3에서 확장하여 bmi 변수를 생성, 이후 키와 몸무게 순으로 입력 받아 해당 변수를 통해 연산한 값을 bmi 변수에 저장했습니다. 이후 if 문을 여러 개 사용하여 조건을 만족하는 경우에 써둔 문장을 출력하게 했습니다. 저체중 bmi를 출력할 때 에러가 뜨지만 아마 형이 정수형이어서 인 것 같습니다. 실수형으로 어떻게 바꾸는지 몰라 부득이하게 오류가 난 상태로 돌렸습니다.

4번

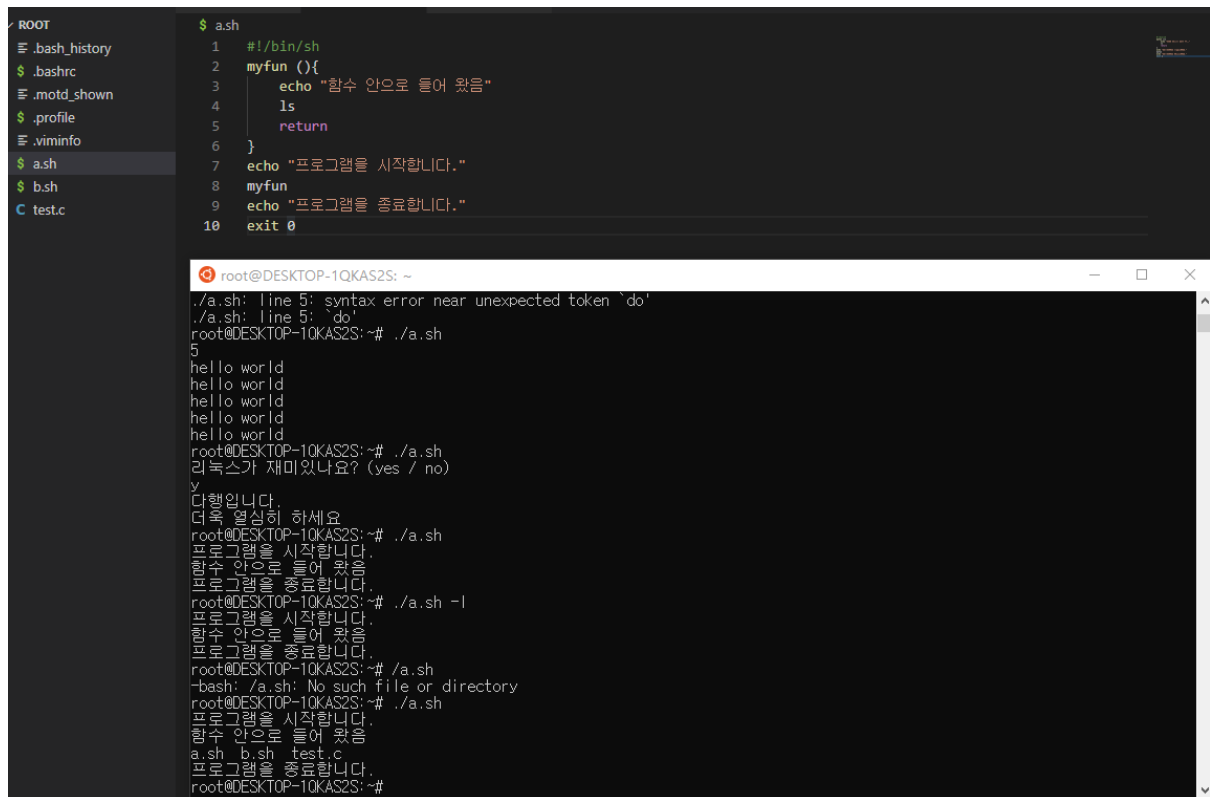
```
a.sh
1  #!/bin/sh
2  echo "리눅스가 재미있나요? (yes / no)"
3  read answer
4  case $answer in
5      yes | y | Y | Yes | YES)
6          echo "다행입니다."
7          echo "더욱 열심히 하세요";;
8      [nN]*)
9          echo "안타깝네요. ππ";;
10     *)
11         echo "yes 아니면 no만 입력했어야죠"
12         exit 1;;
13 esac
14 exit 0
15
```

root@DESKTOP-1QKAS2S: ~

```
./a.sh: line 4: [: missing `]'
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: line 4: [: too many arguments
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: line 4: [: too many arguments
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: line 4: while[ 0 -le 5]: command not found
./a.sh: line 5: syntax error near unexpected token `do'
./a.sh: line 5: `do'
root@DESKTOP-1QKAS2S:~# ./a.sh
5
./a.sh: line 4: while[ 1 -le 5 ]: command not found
./a.sh: line 5: syntax error near unexpected token `do'
./a.sh: line 5: `do'
root@DESKTOP-1QKAS2S:~# ./a.sh
5
hello world
hello world
hello world
hello world
hello world
root@DESKTOP-1QKAS2S:~# ./a.sh
리눅스가 재미있나요? (yes / no)
y
다행입니다.
더욱 열심히 하세요
root@DESKTOP-1QKAS2S:~#
```

교과서에 있는 예제를 따라 실습해보았습니다.

5번



```
ROOT
└─ .bash_history
└─ .bashrc
└─ .motd_shown
└─ .profile
└─ .viminfo
└─ a.sh
└─ b.sh
└─ test.c

$ a.sh
1  #!/bin/sh
2  myfun (){
3      echo "함수 안으로 들어 왔음"
4      ls
5      return
6  }
7  echo "프로그램을 시작합니다."
8  myfun
9  echo "프로그램을 종료합니다."
10 exit 0

root@DESKTOP-1QKAS2S: ~
./a.sh: line 5: syntax error near unexpected token `do'
./a.sh: line 5: `do'
root@DESKTOP-1QKAS2S: ~# ./a.sh
5
hello world
hello world
hello world
hello world
hello world
root@DESKTOP-1QKAS2S: ~# ./a.sh
리눅스가 재미있나요? (yes / no)
y
다행입니다.
더욱 열심히 하세요
root@DESKTOP-1QKAS2S: ~# ./a.sh
프로그램을 시작합니다.
함수 안으로 들어 왔음
프로그램을 종료합니다.
root@DESKTOP-1QKAS2S: ~# ./a.sh -l
프로그램을 시작합니다.
함수 안으로 들어 왔음
프로그램을 종료합니다.
root@DESKTOP-1QKAS2S: ~# ./a.sh
-bash: ./a.sh: No such file or directory
root@DESKTOP-1QKAS2S: ~# ./a.sh
프로그램을 시작합니다.
함수 안으로 들어 왔음
a.sh b.sh test.c
프로그램을 종료합니다.
root@DESKTOP-1QKAS2S: ~#
```

교과서에 예제로 나와있는 함수를 참고해서 ls 명령을 통해 함수가 생성된 뒤 파일이 생성된 것을 확인 했습니다.

6번

```
$ a.sh
1  #!/bin/sh
2  if [ ! -f files.txt ]; then
3
4      touch file0.txt;
5      touch file1.txt;
6      touch file2.txt;
7      touch file3.txt;
8      touch file4.txt;
9      ls
10     xz file0.txt;
11     xz file1.txt;
12     xz file2.txt;
13     xz file3.txt;
14     xz file4.txt;
15 fi
16
1 root@DESKTOP-1QKAS2S: ~
root@DESKTOP-1QKAS2S:~# ./a.sh
a.sh b.sh file0.txt file1.txt file2.txt file3.txt file4.txt test.c
root@DESKTOP-1QKAS2S:~# ./a.sh
a.sh b.sh file0.txt file1.txt file2.txt file3.txt file4.txt test.c
root@DESKTOP-1QKAS2S:~#
```

Touch 와 xz 명령어를 사용해 파일을 생성하고 압축했습니다. 파일을 풀 때 xz-d 명령어를 사용한다는 것을 교과서를 통해 알았습니다만 새로운 파일을 생성한 뒤 어떻게 파일을 풀어야 할지 감이 안 잡혀 할 수 있는 부분까지만 구현해보았습니다.

7번

자료를 찾아보니 for 문을 통해서 난수를 만든 뒤 함수를 생성하는 방법이 있다는 것을 알았습니다만, 문제에 제시된 "입력 받은" 이름으로 함수를 만들어야 한다는 뜻은 입력 받은 함수를 기준으로 일정한 영역 내에서(함수이름이 a이면 a1, a2, a3 이런 식으로) 만들어야 한다는 뜻으로 이해했고 이 부분을 어떻게 해결해야할 지 몰라 코드 구현에 실패했습니다. 죄송합니다.

8번

지난 학기 python에서 배운 것처럼 일정한 파일에 입력 받은 이름과 전화번호를 저장하고 문제 4번처럼 이름의 일부분만 입력하더라도 값이 출력되게 하는 코드를 생성해야 한다는 것은 이해했습니다만 7번에서처럼 생성한 파일에 인자를 어떻게 넣어야 하는지 감이 잡히지 않아 코드 구현에 실패했습니다. 죄송합니다.

9번

8번의 연장선으로 8번에 저장해둔 정보를 파일을 불러내고 함수를 하나 만들어 내서 4번처럼 이름의 일정 알파벳만 있더라도 정보가 출력되게끔 해야 한다는 부분은 이해했습니다만 7번 구현에 실패해 8번 코드 또한 구현에 실패했습니다. 죄송합니다.