

Homework 1-1

General information: Refer to the following codes to answer the given questions.

Problem 1

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int n;
6
7      // Suppose that n is always larger than 0.
8      scanf("%d", &n);
9
10     int a = 0;
11     int b = 1;
12     int next = a + b;
13
14     for(int i = 3; i <= n; i++){
15         a = b;
16         b = next;
17         next = a + b;
18     }
19
20     printf("%d\n", a);
21
22     return 0;
23 }
```

- 1) Explain what this code is doing. 피보나치수열의 구하는 코드이다. n 값을 입력받아 수열의 n 번째항의 값을구한다. a,b 를 초기화후 다음항 next 는 a+b 로 초기화한다.
- 2) What is the output of this code? n 이 1 이면 0, n 이 2 부터는 n 번째 피보나치수열값이 output 으로 출력된다.
- 3) Why do we have to use "stdio.h"? What will happen when we do not use that header file? Stdio.h 헤더파일은 표준입출력 함수를 정의하는 헤더파일이다.

Stdio.h 헤더파일을 사용하지 않으면 컴파일러는 printf/scanf 함수를 알지못하므로 에러가 발생할것입니다.
- 4) Does this code always produce the right answer? If not, when does this code print

the wrong answer? 항상 올바른값을 만들어내지 않는다. $n=1$ 일 때 0 이 출력되는데 피보나치수열의 첫항은 1 이다. 또한 n 의값이 매우 큰값이 주어지는 경우에는 메모리 오버플로우가 발생할 가능성이 있다.

- 5) Make your own code which uses a while statement instead of a for statement.

```
1  #include <stdio.h>
2
3  int main() {
4      int n;
5
6      // Suppose that n is always larger than 0.
7      scanf("%d", &n);
8
9      int a = 0;
10     int b = 1;
11     int next = a + b;
12     int i = 3;
13
14     while (i <= n) {
15         a = b;
16         b = next;
17         next = a + b;
18         i++;
19     }
20
21     printf("%d\n", a);
22
23     return 0;
24 }
```

Problem 2

```
1  #include <stdio.h>
2
3  int main()
4  {
5
6      int flag;
7      int num;
8
9      scanf("%d", &num);
10
11     for(int i = 2; i < num; i++){
12         if(!(num % i)) {
13             flag = 0;
14             break;
15         } else {
16             flag = 1;
17         }
18     }
19
20     switch(flag){
21         case 0:
22             printf("False\n");
23             break;
24         case 1:
25             printf("True\n");
26             break;
27         default:
28             printf("...What?\n");
29     }
30
31     return 0;
32
33 }
```

- 1) Explain what this code is doing.

Prime number 인지 판별하는 코드이다. 소수면 TRUE 아니면 FALSE 를 반환한다.

- 2) If the increment statement of the for loop in line 11 was changed to “++i”, would the code’s behavior change? Explain. 차이가 없다. 왜냐하면 이코드에서 if 문에 사용된 변수 i 는 어차피 값이 증가되기전에 사용되므로 전위증가연산자와 후위증가연산자 어느것을 사용해도 차이가 없다. 연산속도에서는 조금 차이가 있을수있다.
- 3) Is the if-condition in line 12 functional, even though no relational or equality

operators are being used? If yes, explain. If not, correct the condition.

functional 하다. 왜냐하면 나머지가 0 이아니라면 num 이 i 로 나누어 떨어지지않다는것이다. 즉, num 이 소수인지 판별하는데 funtional 합니다.

- 4) Is there any chance that the switch statement's default branch in line 27 will be executed? Why? 가능성이 있다. Num=2 인경우에 for 문의 조건문($i < \text{num}$)이 거짓이되므로 for 문이 실행되지않는다. 그러므로 flag 의값이초기화 되지않았다. 이경우에는 switch 문의 default 가 실행 될수있다.

Problem 3

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int a;
6      scanf("%d", &a);
7
8      if (a >= 90)
9          printf("A\n");
10     else if (a >= 80)
11         printf("B\n");
12     else if (a >= 70)
13         printf("C\n");
14     else if (a >= 60)
15         printf("D\n");
16     else
17         printf("F\n");
18
19     return 0;
20 }
```

- 1) Describe the process and output of this C program.

변수 a 에 정수를 입력받고 a 에따라 성적등급을 출력하는 과정이다. 90 이상은 A, 90 미만 80 이상은 B, 80 미만 70 이상은 C, 70 미만 60 이상은 D, 60 미만은 F 가 출력된다.

- 2) What is the purpose of the variable a? Suggest a more meaningful name for this variable.

사용자가 입력하는 값을 저장하기 위한 목적이다. 성적등급에 관한 것이므로 grade 또는 rank 가 더 의미있을 것이다.


- 3) In line 10, why is the condition of the 'else if' statements is 'a >= 80', instead of 'a >= 80 && a < 90'? Please explain your answer. If your answer is "No", please provide the correct code, too.

a >= 80 과 a >= 80 && a < 90 은 같은 결과이다. 이전 if 와 elseif 문에서 a >= 인 케이스는 이미 적용되어 판별되었기 때문이다.

- 4) Consider whether the if statement could be restructured using a switch statement. If feasible, demonstrate how the code would look after conversion and discuss the pros and cons of using 'switch' over 'if-else'. If a 'switch' statement is not appropriate, please justify why.

If-else 문 대신 switch 문으로 재구성하는 것이 적절하다.

a 가 정수값으로 주어지고 switch 문을 사용하면 각각의 경우들을 case 블록으로 나타내어 가독성이 좋기에 적절하다.



```
1  #include <stdio.h>
2
3  int main() {
4      int a;
5      scanf("%d", &a);
6
7      switch (a / 10) {
8          case 10:
9          case 9:
10         printf("A\n");
11         break;
12         case 8:
13         printf("B\n");
14         break;
15         case 7:
16         printf("C\n");
17         break;
18         case 6:
19         printf("D\n");
20         break;
21         default:
22         printf("F\n");
23         break;
24     }
25
26     return 0;
27 }
```

Problem 4

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int product = 1;
6      int counter = 0;
7      do {
8          product = 3 * product;
9          counter++;
10     } while(product <= 1000); // Increase the limit to 1000
11
12     printf("Final product: %d\n", product);
13     printf("Iterations: %d\n", counter);
14     return 0;
15 }
```

- 1) What is the code doing? What is the result(output) of the code?

product 변수의 값을 3 을 계속 곱하고 counter 변수의 값을 1 씩 증가시키는 루프이다.

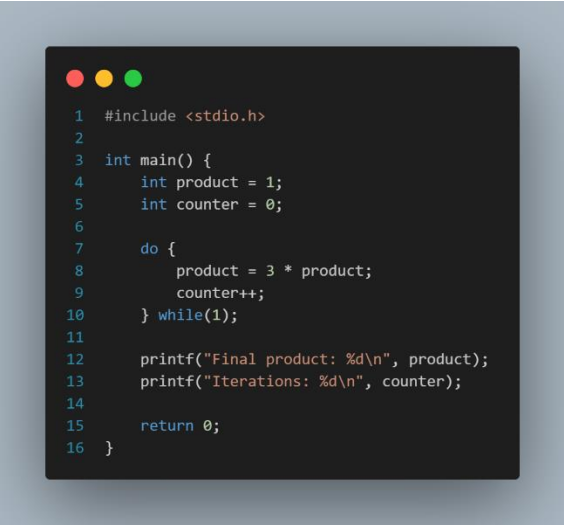
product 의 값이 1000 이하일 때 까지 반복하고 최종으로는 product 의 값과 루프의 실행횟수를 print 해준다. 위코드의 결과는

Final product: 2187

Iterations: 7

이다.

- 2) Rewrite the code to be an indefinite loop using a do...while loop



```
1  #include <stdio.h>
2
3  int main() {
4      int product = 1;
5      int counter = 0;
6
7      do {
8          product = 3 * product;
9          counter++;
10     } while(1);
11
12     printf("Final product: %d\n", product);
13     printf("Iterations: %d\n", counter);
14
15     return 0;
16 }
```

3) Rewrite the code to use a while loop instead of a do..while loop

```
1  #include <stdio.h>
2
3  int main() {
4      int product = 1;
5      int counter = 0;
6
7      while(product <= 1000) {
8          product = 3 * product;
9          counter++;
10     }
11
12     printf("Final product: %d\n", product);
13     printf("Iterations: %d\n", counter);
14
15     return 0;
16 }
```

4) Rewrite the code to use break with a while loop

```
1  #include <stdio.h>
2
3  int main() {
4      int product = 1;
5      int counter = 0;
6
7      while(1) {
8          product = 3 * product;
9          counter++;
10         if(product > 1000)
11             break;
12     }
13
14     printf("Final product: %d\n", product);
15     printf("Iterations: %d\n", counter);
16
17     return 0;
18 }
```

5) What will happen if the 8th line and 9th line are changed? What will be the result(output) of the code?

Final product: 2187

Iterations: 7

값이 바뀌지않는다. 두줄의 실행순서는 바뀌지만 루프에서 product 의변수가 바뀌는 방식이나 루프의 종료조건도 동일하기때문이다.