HW #1

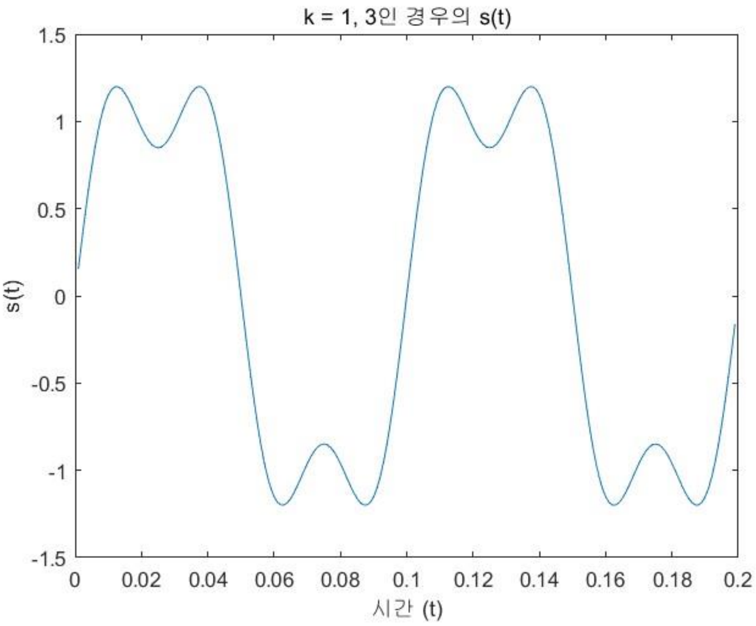
그래프와 소스 코드에 대하여

과목명: 데이터통신
교수: 최승식
학과: 컴퓨터공학부
학번: 202201479
이름: 박지원

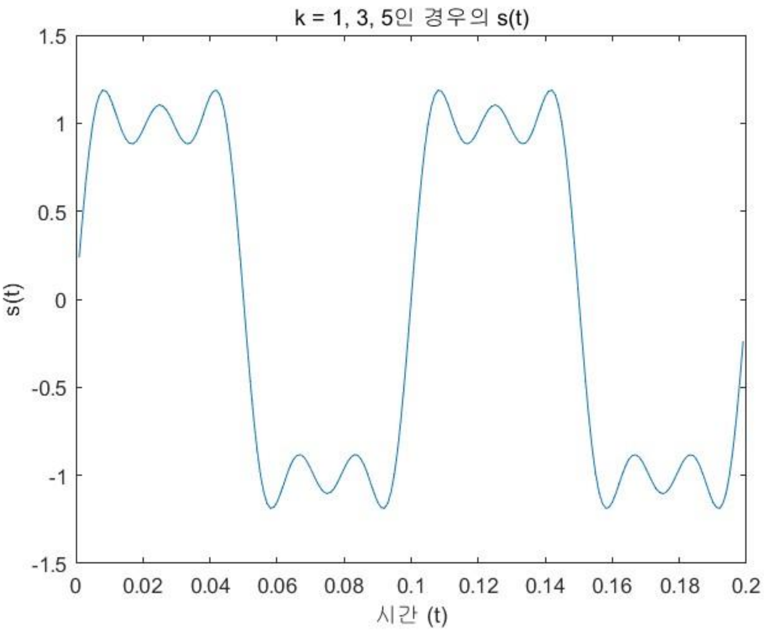
목차

Ⅰ. 문제 1번 _ ⅰ. 그래프

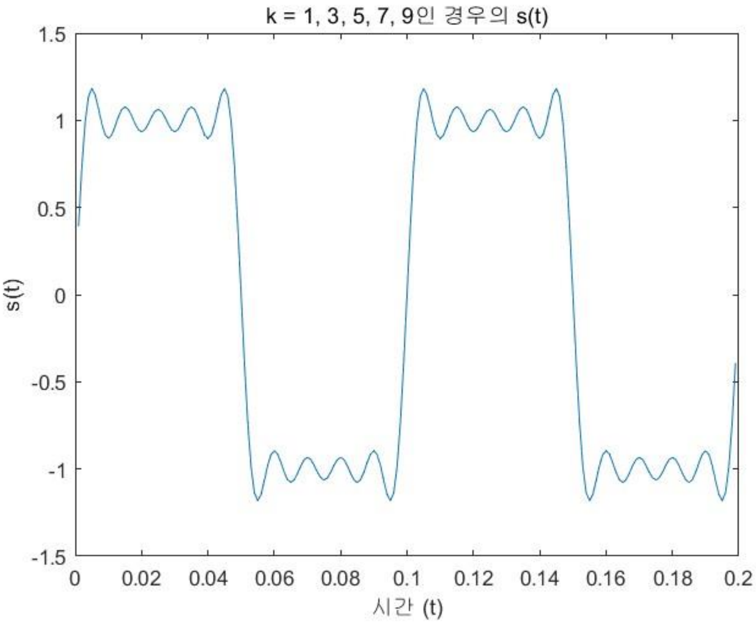
a. k = 1, 3 인 경우에 s(t) 그래프



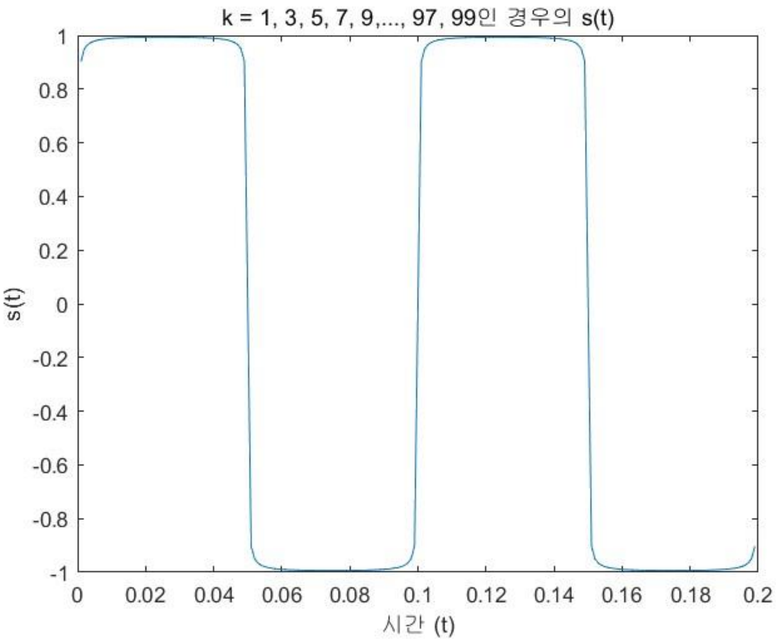
b. k = 1, 3, 5 인 경우에 s(t) 그래프



c. k = 1, 3, 5, 7, 9 인 경우에 s(t) 그래프



d. k = 1, 3, 5, 7..., 97, 99 인 경우에 s(t) 그래프



.

I. 문제 1번 _ ii. 코드 작성

a. k = 1, 3 인 경우에 s(t) 그래프

```
% 주어진 변수 설정
A = 1;
f = 10;
t = 0:0.001:0.2; % 0 부터 0.2 까지 0.001 간격으로
t = t(t > 0 & t < 0.2); % 0 초 초과 0.2 초 미만 필터링


% (a) k = 1,3 인 경우
n = [1, 3];
s = zeros(size(t));
for k = n
    s = s + (4/pi) * sin(2*pi*k*f*t)/k;
end
figure; % 새로운 그래프 창을 연다
plot(t, s);
title('k = 1, 3 인 경우의 s(t)');
xlabel('시간 (t)');
ylabel('s(t)');
```

b. k = 1, 3, 5 인 경우에 s(t) 그래프

```
% 주어진 변수 설정
A = 1;
f = 10;
t = 0:0.001:0.4; % 0 부터 0.4 까지 0.001 간격으로
t = t(t > 0 & t < 0.4); % 0 초 초과 0.4 초 미만 필터링
% (b) k = 1,3,5 인 경우
n = [1, 3, 5];
s = zeros(size(t));
for k = n
    s = s + (4/pi) * sin(2*pi*k*f*t)/k;
end
figure;
plot(t, s);
title('k = 1, 3, 5 인 경우의 s(t)');
xlabel('시간 (t)');
ylabel('s(t)');
```

c. k = 1, 3, 5, 7, 9 인 경우에 s(t) 그래프

```
% 주어진 변수 설정
A = 1;
f = 10;
t = 0:0.001:0.4; % 0 부터 0.4 까지 0.001 간격으로
t = t(t > 0 & t < 0.4); % 0 초 초과 0.4 초 미만 필터링
% (c) k = 1,3,5,7,9 인 경우
n = [1, 3, 5, 7, 9];
s = zeros(size(t));
for k = n
    s = s + (4/pi) * sin(2*pi*k*f*t)/k;
end
figure;
plot(t, s);
title('k = 1, 3, 5, 7, 9 인 경우의 s(t)');
xlabel('시간 (t)');
ylabel('s(t)');
```

d. k = 1, 3, 5, 7, ... , 97, 99 인 경우에 s(t) 그래프

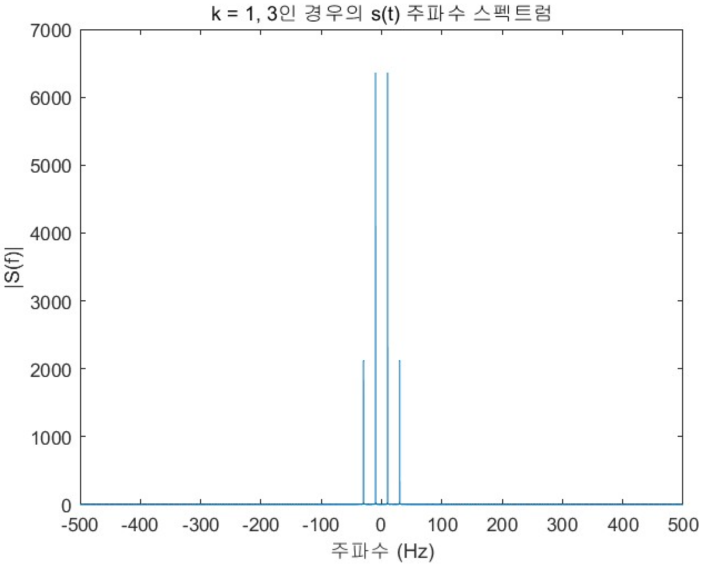
```
% 주어진 변수 설정
A = 1;
f = 10;
t = 0:0.001:0.4; % 0 부터 0.4 까지 0.001 간격으로
t = t(t > 0 & t < 0.4); % 0 초 초과 0.4 초 미만 필터링
% (d) k = 1,3,5,7,9,...,97,99 인 경우
n = 1:2:99;
s = zeros(size(t));
for k = n
    s = s + (4/pi) * sin(2*pi*k*f*t)/k;
end
figure;
plot(t, s);
title('k = 1, 3, 5, 7, 9,..., 97, 99 인 경우의 s(t)');
xlabel('시간 (t)');
ylabel('s(t)');
```

Ⅰ. 문제 1 번 _ iii. 문제점

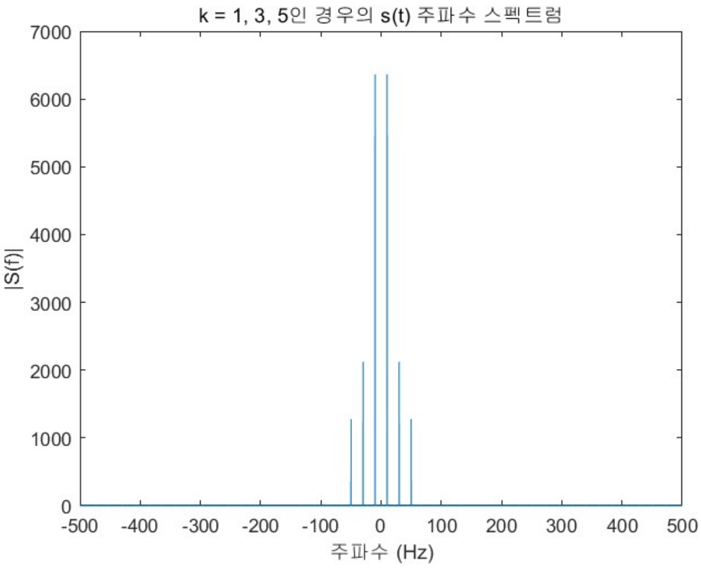
위 코드들의 문제점은 결국 근사화 값이라는 점이다. 코드에서 보면 알다시피, 주어진 s(t)의 식을 그대로 사용하는 것이 아니라, s = s + (4/pi) * sin(2*pi*k*f*t)/k; 라는 식을 적절한 k 값의 범위에 맞춰 반복하여 합한다. 이런 식으로 근사값을 구한 이유는 주어진 s(t)의 식에서 A 와 4/pi 값은 아주 큰 범위에서 보면 결과 값에 큰 영향을 미치지 않기 때문이다.

Ⅱ. 문제 2번 _ ⅰ. 그래프

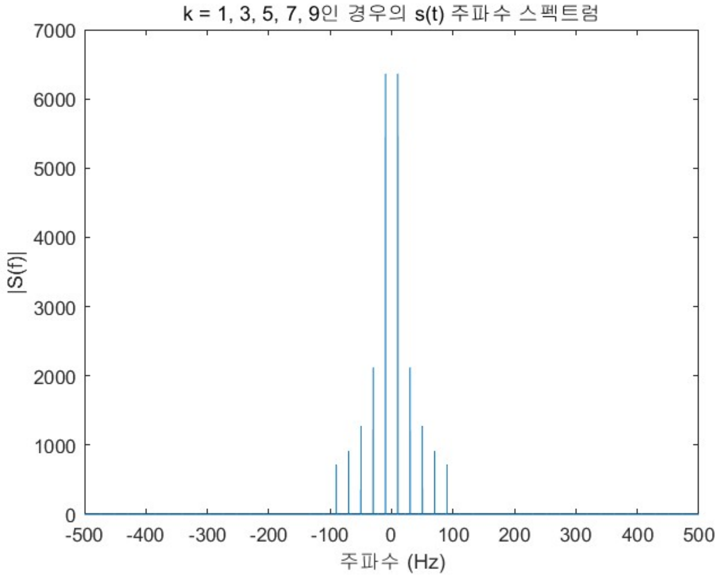
a. k = 1, 3 인 경우 주파수 스펙트럼



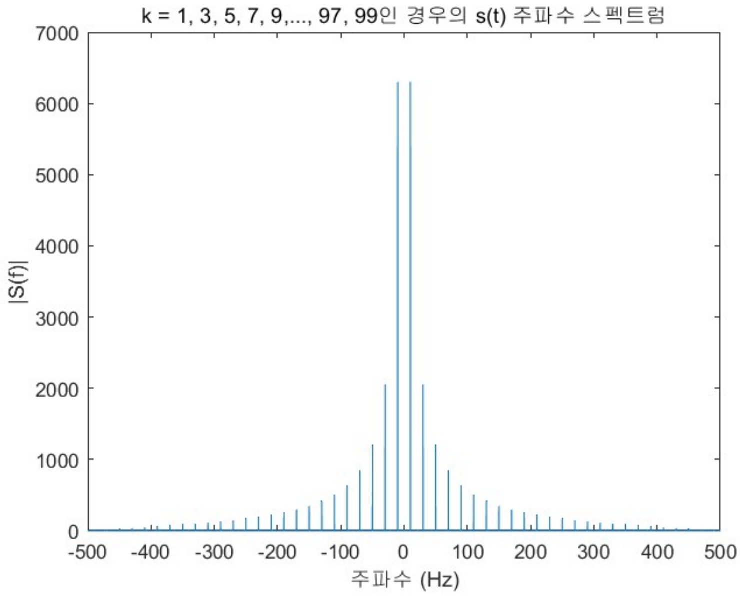
b. k = 1, 3, 5 인 경우 주파수 스펙트럼



c. k = 1, 3, 5, 7, 9 인 경우 주파수 스펙트럼



d. k = 1, 3, 5, 7, 9 , … ,97, 99 인 경우 주파수 스펙트럼



Ⅱ. 문제 2번 _ ii. 코드 작성 과정

a. k = 1, 3 인 경우 주파수 스펙트럼

```
% 주어진 변수 설정
A = 1;
f = 10;
ts = 1/1000; % 샘플링 간격
t = 0:ts:10; % 0부터 10까지 ts 간격으로
t = t(t > 0 & t < 10);

% k = 1, 3인 경우
n = [1, 3];
s = zeros(size(t));
for k = n
    s = s + (4./pi) .* sin(2.*pi.*k.*f.*t)./k;
end

% FFT 계산
y = fft(s);

% FFT 결과를 중심으로 이동
yshift = fftshift(y);

% 주파수 축 계산
N = length(t); % 샘플의 개수
df = 1/(N*ts); % 주파수 해상도
f = -1/(2*ts) : df : 1/(2*ts)-df; % 주파수 벡터 생성

% 주파수 스펙트럼 그리기
figure;
plot(f, abs(yshift));
title('k = 1, 3인 경우의 s(t) 주파수 스펙트럼');
xlabel('주파수 (Hz)');
ylabel('|S(f)|');
```

b. k = 1, 3 인 경우 주파수 스펙트럼

```
% 주어진 변수 설정
A = 1;
f = 10; % 기본 주파수
ts = 1/1000; % 샘플링 간격
t = 0:ts:10-ts; % 0 부터 10 까지 ts 간격으로, 10 을 포함하지 않음

% k = 1, 3, 5 인 경우의 s(t) 함수 생성
n = [1, 3, 5];
s = zeros(size(t));
for k = n
    s = s + (4/pi) * sin(2*pi*k*f*t)/k;
end

% FFT 계산
y = fft(s);

% FFT 결과를 중심으로 이동
yshift = fftshift(y);

% 주파수 축 계산
N = length(t); % 샘플의 개수
df = 1/(N*ts); % 주파수 해상도
fAxis = -1/(2*ts) : df : 1/(2*ts)-df; % 주파수 벡터 생성

% 주파수 스펙트럼 그리기
figure;
plot(fAxis, abs(yshift));
title('k = 1, 3, 5 인 경우의 s(t) 주파수 스펙트럼');
xlabel('주파수 (Hz)');
ylabel('|S(f)|');
```

c. k = 1, 3 인 경우 주파수 스펙트럼

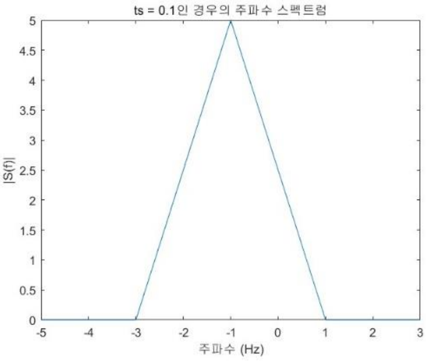
```
% 주어진 변수 설정
A = 1;
f = 10; % 기본 주파수
ts = 1/1000; % 샘플링 간격
t = 0:ts:10-ts; % 0 부터 10 까지 ts 간격으로, 10 을 포함하지 않음

% k = 1, 3, 5, 7, 9 인 경우의 s(t) 함수 생성
n = [1, 3, 5, 7, 9];
s = zeros(size(t));
for k = n
    s = s + (4/pi) * sin(2*pi*k*f*t)/k;
end

% FFT 계산
y = fft(s);

% FFT 결과를 중심으로 이동
yshift = fftshift(y);

% 주파수 축 계산
N = length(t); % 샘플의 개수
df = 1/(N*ts); % 주파수 해상도
fAxis = -1/(2*ts) : df : 1/(2*ts)-df; % 주파수 벡터 생성

% 주파수 스펙트럼 그리기
figure;
plot(fAxis, abs(yshift));
title('k = 1, 3, 5, 7, 9 인 경우의 s(t) 주파수 스펙트럼');
xlabel('주파수 (Hz)');
ylabel('|S(f)|');
```

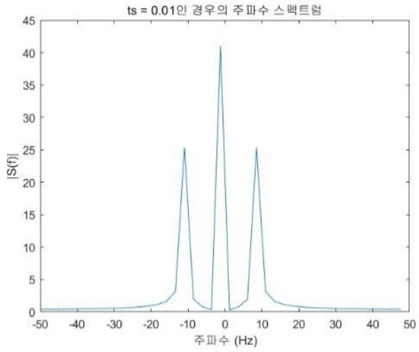
d. k = 1, 3 인 경우 주파수 스펙트럼

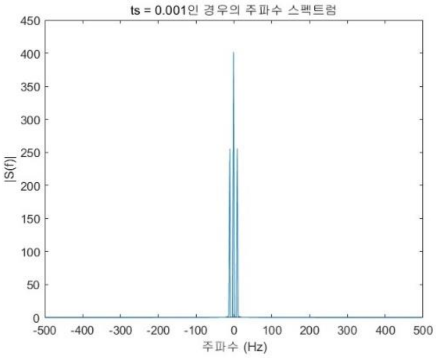
```
% 주어진 변수 설정
f = 10; % 기본 주파수
ts = 1/1000; % 샘플링 간격
t = 0:ts:10-ts; % 0 부터 10 까지 ts 간격으로, 10 을 포함하지 않음

% k = 1, 3, 5, 7, 9,..., 97, 99 인 경우의 s(t) 함수 생성
n = 1:2:99;
s = zeros(size(t));
for k = n
    s = s + (4/pi) * sin(2*pi*k*f*t)/k;
end

% FFT 계산
y = fft(s);

% FFT 결과를 중심으로 이동
yshift = fftshift(y);

% 주파수 축 계산
N = length(t); % 샘플의 개수
df = 1/(N*ts); % 주파수 해상도
fAxis = -1/(2*ts) : df : 1/(2*ts)-df; % 주파수 벡터 생성

% 주파수 스펙트럼 그리기
figure;
plot(fAxis, abs(yshift));
title('k = 1, 3, 5, 7, 9,..., 97, 99 인 경우의 s(t) 주파수 스펙트럼');
xlabel('주파수 (Hz)');
ylabel('|S(f)|');
```

Ⅲ. 문제 3번 _ i . 그래프

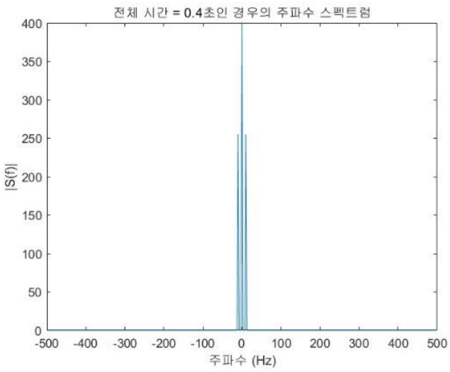
1) 함수 샘플링 간격 변화

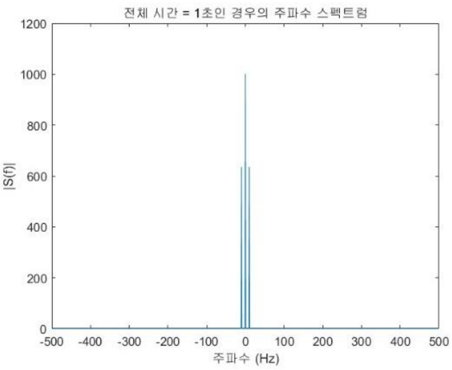


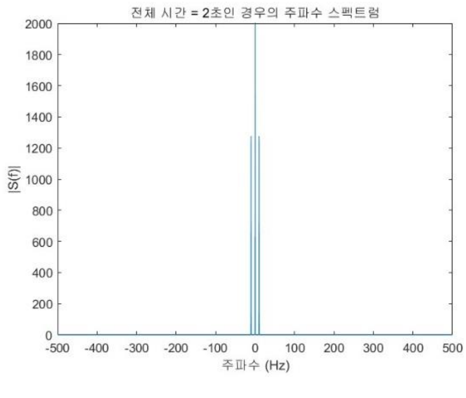




2) 입력되는 함수의 전체 시간 변화







Ⅲ. 문제 3번 _ ii. 코드 작성

1) 함수 샘플링 간격 변화

```
% 변수 설정
A = 1;
f = 10;

% 샘플링 간격 변화 예시
tsArray = [0.001, 0.01, 0.1]; % 다양한 샘플링 간격

for ts = tsArray
    t = 0:ts:0.4; % 주어진 ts 로 시간 벡터 재설정
    s = A + (4/pi) * sin(2*pi*1*f*t)/1; % 간략화된 s(t) 함수, k=1 로 설정

    % FFT 계산
    y = fft(s);
    yshift = fftshift(y);

    % 주파수 축 계산
    N = length(t);
    df = 1/(N*ts);
    fAxis = -1/(2*ts) : df : 1/(2*ts)-df;

    % 주파수 스펙트럼 그리기
    figure;
    plot(fAxis, abs(yshift));
    title(['ts = ', num2str(ts), '인 경우의 주파수 스펙트럼']);
    xlabel('주파수 (Hz)');
    ylabel('|S(f)|');
end
```

2) 입력되는 함수의 전체 시간 변화

```
% 변수 설정
A = 1;
f = 10;
ts = 0.001; % 샘플링 간격 고정

% 전체 시간 변화 예시
tArray = [0.4, 1, 2]; % 다양한 전체 시간

for totalT = tArray
    t = 0:ts:totalT-ts; % 주어진 전체 시간으로 시간 벡터 재설정
    s = A + (4/pi) * sin(2*pi*1*f*t)/1; % 간략화된 s(t) 함수, k=1 로 설정

    % FFT 계산
    y = fft(s);
    yshift = fftshift(y);

    % 주파수 축 계산
    N = length(t);
    df = 1/(N*ts);
    fAxis = -1/(2*ts) : df : 1/(2*ts)-df; % 수정: 주파수 축 계산 완성

    % 주파수 스펙트럼 그리기
    figure;
    plot(fAxis, abs(yshift));
    title(['전체 시간 = ', num2str(totalT), '초인 경우의 주파수 스펙트럼']);
    xlabel('주파수 (Hz)');
    ylabel('|S(f)|');
end
```

IV. 마무리

1번 문제에서 보이는 모든 그래프는 주기적 복합 신호이다. 크게 보면 sin 형태이고, k값이 늘어날 수 록 세부적으로 보이는 그래프가 많음을 알 수 있다. 1번 문제에서 그려본 그래프들의 주파수 스펙트럼을 2번 문제에서 그려보았다. S(t)의 그래프가 모두 주기적 복합 신호였기 때문에 이들의 주파수 스펙트럼은 모두 비주기적 신호이다. k값이 많아질 수 록 스펙트럼의 넓이는 넓어짐을 알 수 있다.

3번 문제 중 그 1번은 함수 샘플링의 간격을 변화시켰다. ts가 줄어들 수 록, 주파수의 간격은 좁아지고 |s(f)|의 값은 커짐을 그래프를 통해 확인할 수 있었다. 3번 문제 중 그 2번은 입력되는 함수의 전체 시간을 변화시켰다. 전체 시간이 늘어날 수 록 |S(f)|의 값이 늘어남을 마찬가지로 그래프를 통해 확인할 수 있다.

이렇게 수업시간에 배운 주기적 복합신호와 그에 따른 주파수 영역을 구분해 볼 수 있어 의미 있는 과제였다. 그리고 직접 샘플링의 간격과 함수의 전체 시간을 조절하니, 이들이 각각 어떤 영향을 미치는지 한 눈에 확인할 수 있었다.