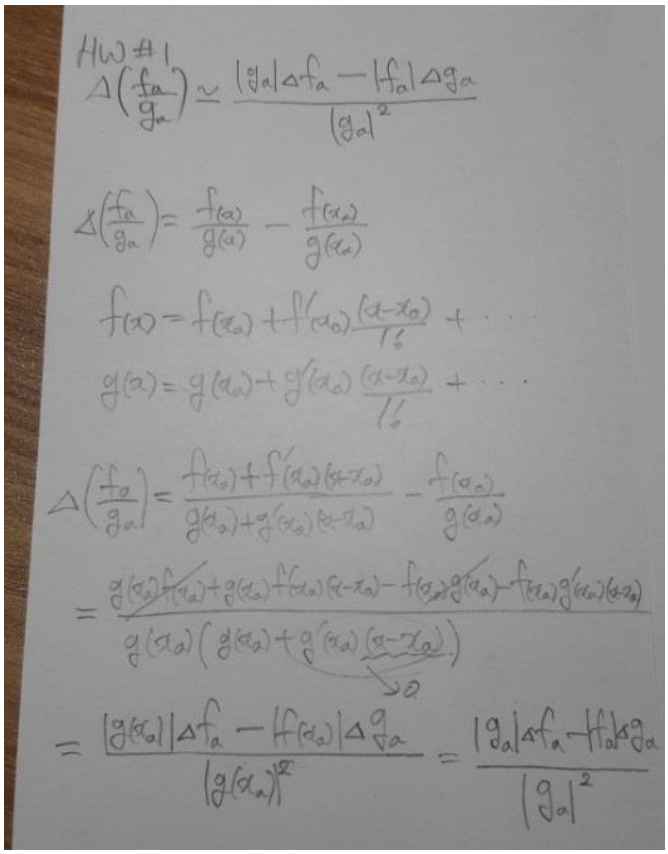
Numerical Methods HW #01

0. Show that $\Delta(\frac{f_a}{g_a}) \cong \frac{|g_a|\Delta f_a - |f_a|\Delta g_a}{|g_a|^2}$

<Solution>



1. 아래 함수의 도함수를 주어진 점에 대해서 도함수의 정의를 이용하여 소수점 5자리까지 정확히 구하라.

(a) $f(x) = e^x \sin x + \cos^2 x$, $x = \frac{\pi}{4}$

<Solution>

도함수의 정의에 의해 $f'(x) = \lim_{h\to 0} \frac{f(x+h)-f(x)}{h}$ 이고 h가 충분히 작으면 $\frac{f(x+h)-f(x)}{h}$로 근사할 수 있다.

구간 h를 1, 1/2, 1/4, … 로 점점 작게 해서 값을 구했다.

구간이 클 때는 error 값이 크지만 구간이 작아질수록 error값이 0에 근접한다.

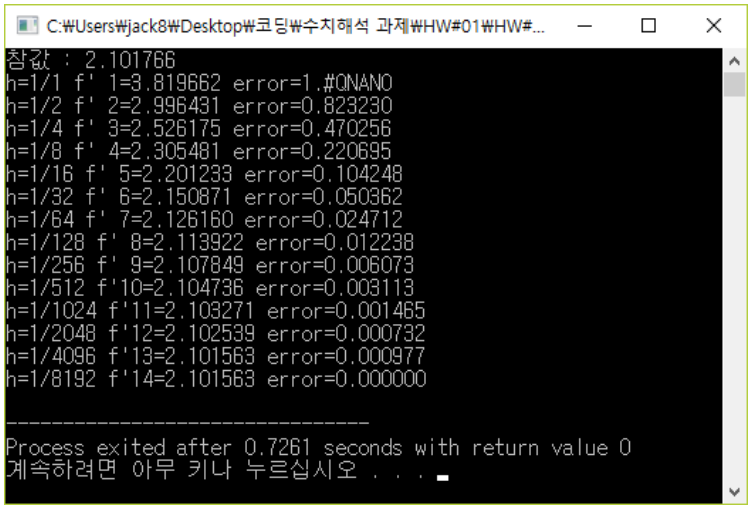
코드에서 break를 걸지 않고 계속 계산해보면 총수치오차에 의해서 오차가 진동하며 튄다.

error값은 Scarborough 판정법에 의해 0.5×10^{-5}로 정했다.

<Code>

```c
#include<stdio.h>
#include<math.h>
float f(float x)
{
	return exp(x)*sin(x)+pow(cos(x),2);
}
int main()
{
	int i;
	double error, a, b,c,d;
	double x=M_PI/4;
	error=0.5/pow(10,5);
	printf("참값 : %lf\n",exp(x)*(sin(x)+cos(x))-2*sin(x)*cos(x));
	for(i=0;i<=100000;i++)
	{
		a=f(x+1/pow(2,i))/(1/pow(2,i));
		b=f(x)/(1/pow(2,i));
		c=a-b;
		printf("h=1/%.0lf f'%2d=%lf error=%lf\n",pow(2,i),i+1,c,fabs(d-c));
		if(i!=1)
		{
			if(fabs(d-c)<error)
			break;
		}
		d=c;
	}
	return 0;
}
```

<결과 창>



첫 번째 시행에서의 error값은 코드에서 d변수에 값이 아직 저장되지 않아 저렇게 표시된다.
컴퓨터를 통해 얻은 값이 참값과 소수점 셋째자리까지는 같지만 그 이후는 다르다.

참 값 =2.101766
계산 값=2.101563

(b) $f(x) = (\ln x)\sin^2 x \quad , \quad x = \frac{\pi}{4}$

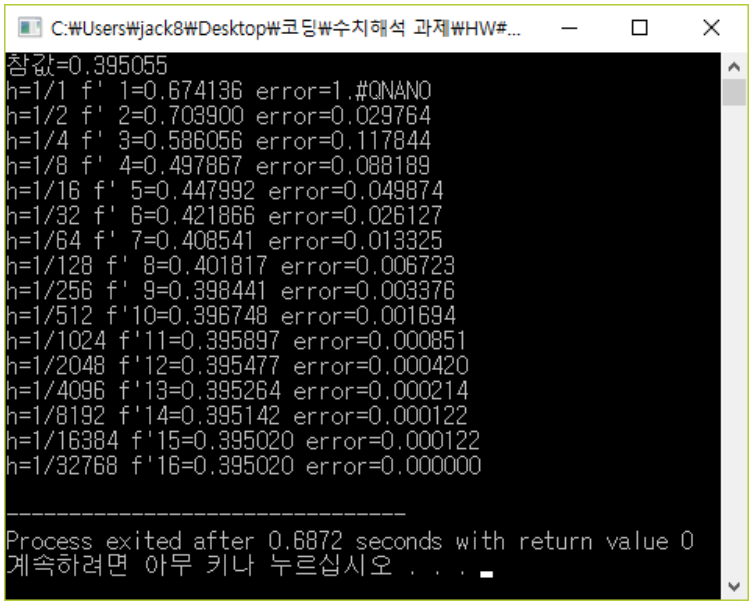
<Solution>

1번 문제의 (a)와 마찬가지로 도함수정의를 구간이 매우 작을 때 $\frac{f(x+h)-f(x)}{h}$로 근사하여 푼다.

<Code>

```
#include<stdio.h>
#include<math.h>
float f(float x)
{
	return log(x)*pow(sin(x),2);
}
int main()
{
	int i;
	double x = M_PI/4;
	double a, error, b;
	error = 0.5/pow(10,6);
	printf("참값=%lf\n",pow(sin(x),2)/x+log(x)*2*sin(x)*cos(x));
	for(i=1;i<=100;i++)
	{
		a=(f(x+(1/pow(2,i-1)))-f(x))/(1/pow(2,i-1));
		printf("h=1/%.0lf f'%2d=%lf error=%lf\n",pow(2,i-1),i,a,fabs(a-b));
		if(i!=1)
		{
			if(fabs(a-b)<error)
			break;
		}
		b=a;
	}
}
	return 0;
}
```

<결과 창>



(a)와 마찬가지로 구간h을 줄여가며 계산을 했고 첫 번째의 error값은 코드에서 b에 저장된 값이 없기 때문에 나온 것이다.
참값과 소수점 4째자리까지는 같지만 5째자리부터는 다르다.

참 값 =0.395055
계산 값=0.395020

2. cos x의 Maclaurin 급수는 다음과 같다.

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots$$

위의 무한급수를 이용하여 $\cos(\frac{\pi}{4})$의 근사값을 유효숫자 3자리까지 정확히 구하라.

<Solution>

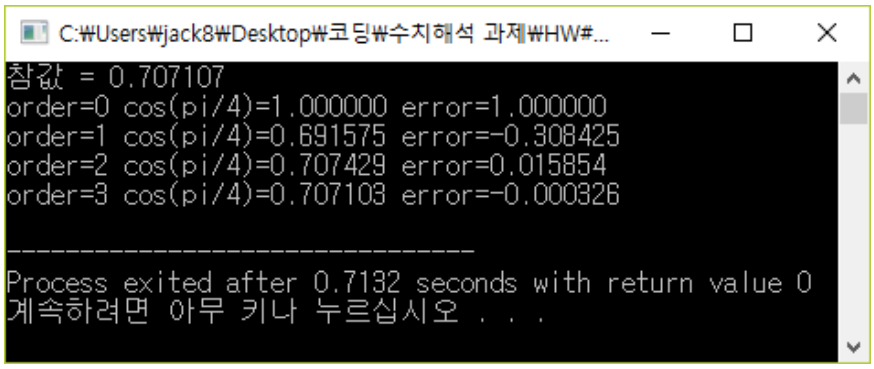
코딩을 통해서 항을 하나씩 추가하는 방법으로 계산했다.

error값은 Scarborough 판정법에 의해 0.5×10^{-3}으로 정했다.

<Code>

```
#include<stdio.h>
#include<math.h>
float factorial(float n)
{
	int i,x;
	x=1;
	if(i=0)
	{
		return 1;
	}
	else
	{
		for(i=1;i<=n;i++)
		{
			x=x*i;
		}
		return x;
	}
}
int main()
{
	int i;
	double x,b,c,error;
	double sum=0;
	x=M_PI/4;
	error=0.5/pow(10,3);
	printf("참값 = %lf\n",cos(x));
	for(i=0;i<=100;i++)
	{
		c=pow(-1,i)*pow(x,2*i)/factorial(2*i);
		sum=sum+c;
		printf("order=%d cos(pi/4)=%lf error=%lf\n",i,sum,c);
		if(i!=1)
		{
			if(fabs(c)<error)
			break;
		}
	}
}
```

<결과 창>



소수점 셋째자리까지 정확히 구하였다.

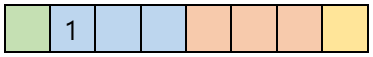
참 값 =0.707107
계산 값=0.707103

3. 버림 작업을 하는 8비트 워드 컴퓨터가 있다. 이 때 가수를 위한 비트 수는 4개이고, 지수를 위한 비트 수는 4개이다. 계산 값이 다음과 같을 때 컴퓨터에서의 저장 값을 구하라.

(a) 1.1135 (b) 0.9501 (c) 7.7532
(d) 7.3243 (e) 0.0523 (f) 0.0685

<Solution>

8비트 워드 컴퓨터이고 가수 비트 4개 지수 비트 4개가 할당되어 있다. 가수와 지수의 부호에 대한 것도 포함한다고 하면 가수를 '표현하기 위한' 비트는 3개, 지수를 표현하기 위한 비트도 3개인 것이다.



왼쪽부터 가수의 부호, 가수, 지수, 지수의 부호를 위한 비트이다.

(a) 1.1135

$$1.1135 = 1 + 0.1135 = 1 + 0 \times \frac{1}{2} + 0 \times \frac{1}{4} + 0.1135\,(\text{버림}) = (0.100)_2 \times 2^{-1} = (01000011)_2$$

01000011

(b) 0.9501

$$0.9501 = 1 \times \frac{1}{2} + 0.4501 = 1 \times \frac{1}{2} + 1 \times \frac{1}{4} + 0.2001 = 1 \times \frac{1}{2} + 1 \times \frac{1}{4} + 1 \times \frac{1}{8} + 0.0751(\text{버림})$$
$$= (0.111)_2 \times 2^0 = (01000000)_2$$

01000000

(c) 7.7532

$$7.7532 = 1 \times 4 + 1 \times 2 + 1 \times 1 + 0.7532\,(\text{버림}) = (0.111)_2 \times 2^3 = (01111110)_2$$

01111110

(d) 7.3243

$$7.3243 = 1 \times 4 + 1 \times 2 + 1 \times 1 + 0.3243\,(\text{버림}) = (0.111)_2 \times 2^3 = (01111110)_2$$

01111110

(e) 0.0523

$$0.0523 \times 2^5 = 1.6736 = 1 \times 1 + 1 \times \frac{1}{2} + 0 \times \frac{1}{4} + 0.1736(\text{버림}) = (0.110)_2 \times 2^1$$
$$0.0523 = (0.110)_2 \times 2^{1-5} = (01101001)_2$$

01101001

(f) 0.0685

$$0.0685 \times 2^5 = 2.192 = 1 \times 2 + 0.192 = 1 \times 2 + 1 \times \frac{1}{8} + 0.067\,(\text{버림}) = (0.100)_2 \times 2^2 = (01000100)_2$$

01000100

4. sin(x)는 다음과 같이 무한급수로 정의된다.

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

이 급수를 사용하여 $\sin\left(\frac{\pi}{4}\right)$을 계산하는 알고리즘을 작성하라. 단, 무한급수에서 근사값에 대한 백분율 상대오차(ε_a)를 각각 출력하되, 그것이 0.005% 이하이면 계산을 중지하라.

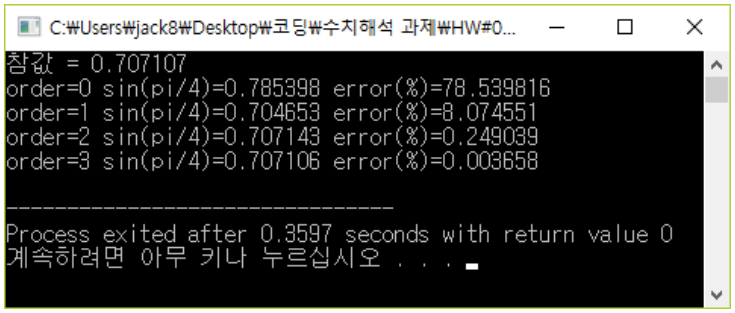
<Solution>

2번 문제와 마찬가지로 항을 하나씩 추가해가면서 계산하는 프로그램을 만들었다.

<Code>

```
#include<stdio.h>
#include<math.h>
float factorial(float n)
{
	int i,x;
	x=1;
	if(i=0)
	{
		return 1;
	}
	else
	{
		for(i=1;i<=n;i++)
		{
			x=x*i;
		}
		return x;
	}
}
int main()
{
	double x,s,error;
	double sum=0;
	int i;
	x=M_PI/4;
	error=0.5/pow(10,4);
	printf("참값 = %lf\n",sin(x));
	for(i=0;i<=100;i++)
	{
		s=pow(-1,i)*pow(x,2*i+1)/factorial(2*i+1);
		sum=sum+s;
		printf("order=%d sin(pi/4)=%lf error(%%)=%lf\n",i,sum,fabs(s)*100);
		if(i!=1)
		{
			if(fabs(s)<error)
			break;
		}
	}
}
```

<결과 창>



소수점 셋째자리까지 정확하게 구하였다.

참 값 =0.707107
계산 값=0.707106

5. 다음 무한급수

$$y = e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$

를 이용하여 e^9와 e^{-9}을 계산하라. 그 결과를 참값과 비교하고 만일 그 결과가 서로 일치하지 않으면 그 이유를 서술하라. 단 e^9와 e^{-9}의 참값은 8103.08과 1.23409×10^{-4} 이다.

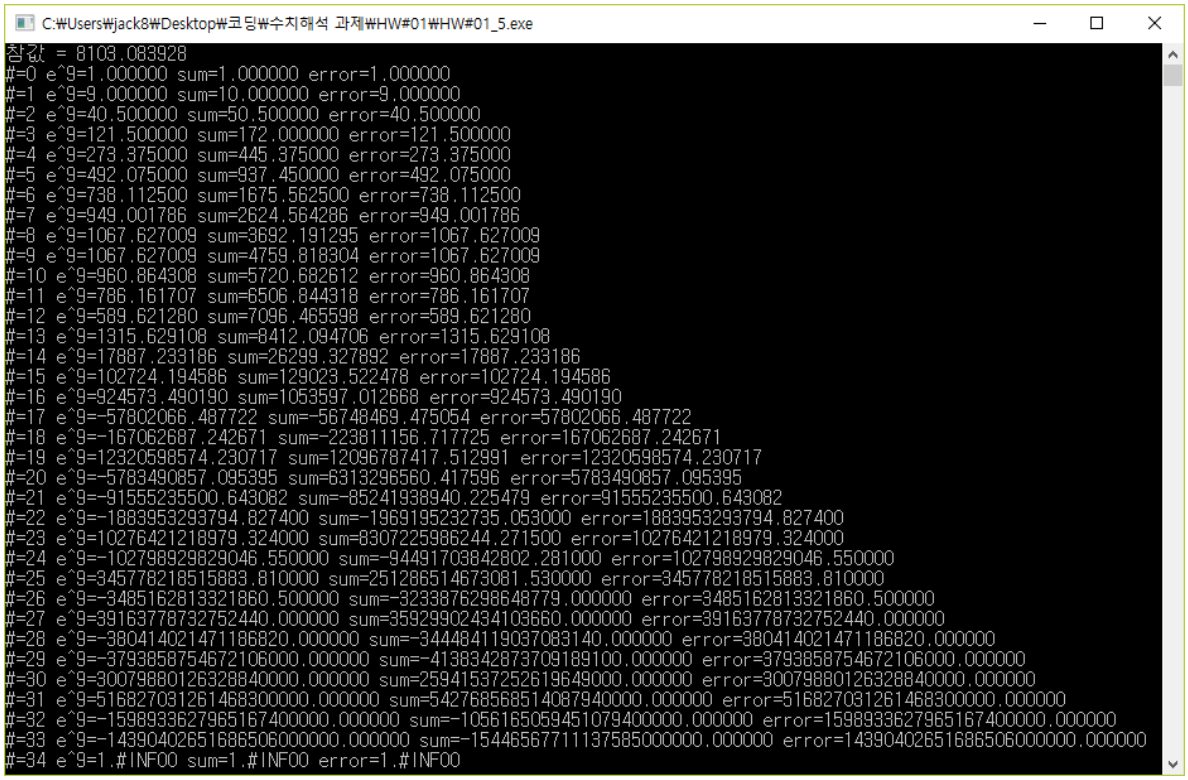
<Solution>

2번이나 4번문제처럼 코드를 짰을 때는 아래 결과창과 같이 14번시행부터 오차가 갑자기 커졌다. 항을 하나씩 더할 때 분자와 분모를 따로 계산해서 더하면 분자와 분모가 매우 커져 계산이 안되는 것을 보고 하나의 항을 만들 때 (x/n)을 n은 1부터 n까지 곱해서 만들었다.

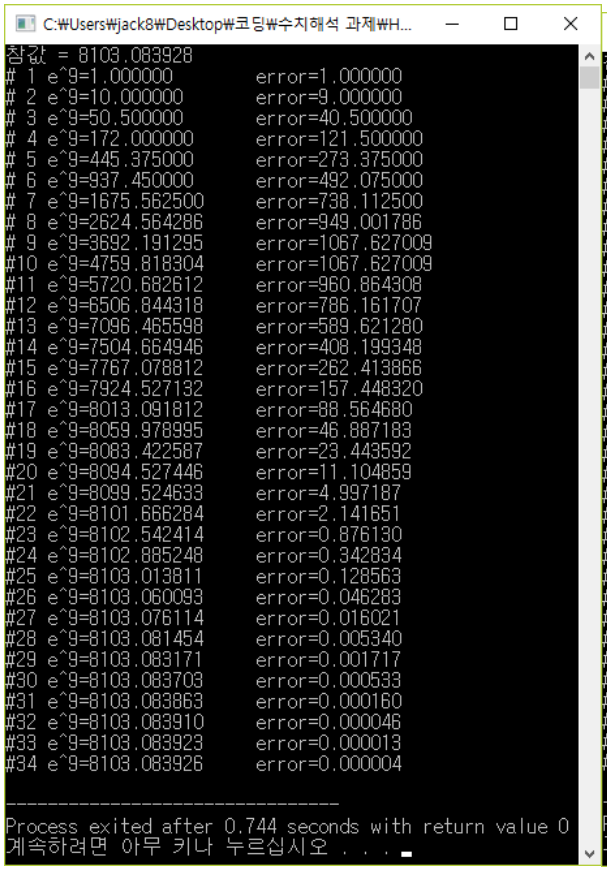
<Code>

```
#include<stdio.h>
#include<math.h>
int main()
{
	int i;
	double j;
	double x=9;
	double e,sum,f;
	double error=0.5/pow(10,5);
	sum=0;
	printf("참값 = %lf\n",pow(M_E,x));
	for(i=0;i<=100;i++)
	{
		f=1;
		if(i==0)
		{
			f=1;
		}
		else
		{
			for(j=1;j<=i;j++)
			{
				e=x/j;
				f=f*e;
			}
		}
		sum=sum+f;
		printf("#%2d e^%d=%lf error=%lf\n",i+1,(int)x,sum,f);
		if(fabs(f)<error) break;
	}
}
```

<결과 창>



2번과 4번 문제처럼 항을 하나씩 더해서 얻은 결과



항을 (x/n)을 곱해가며 만들어서 얻은 결과

e^{9}의 참값=8103.08, 계산값=8103.083926 e^{-9}의 참값=1.23409×10^{-4} 계산값=0.000123