In [1]:
```
from sklearn.datasets import load_breast_cancer
import pandas as pd

# 데이터 로드
cancer = load_breast_cancer()

# DataFrame으로 변환 (타겟 값 제외)
data = pd.DataFrame(cancer.data, columns=cancer.feature_names)
```

In [3]:
```
# 데이터 정보 확인
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 30 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   mean radius              569 non-null    float64
 1   mean texture             569 non-null    float64
 2   mean perimeter           569 non-null    float64
 3   mean area                569 non-null    float64
 4   mean smoothness          569 non-null    float64
 5   mean compactness         569 non-null    float64
 6   mean concavity           569 non-null    float64
 7   mean concave points      569 non-null    float64
 8   mean symmetry            569 non-null    float64
 9   mean fractal dimension   569 non-null    float64
 10  radius error             569 non-null    float64
 11  texture error            569 non-null    float64
 12  perimeter error          569 non-null    float64
 13  area error               569 non-null    float64
 14  smoothness error         569 non-null    float64
 15  compactness error        569 non-null    float64
 16  concavity error          569 non-null    float64
 17  concave points error     569 non-null    float64
 18  symmetry error           569 non-null    float64
 19  fractal dimension error  569 non-null    float64
 20  worst radius             569 non-null    float64
 21  worst texture            569 non-null    float64
 22  worst perimeter          569 non-null    float64
 23  worst area               569 non-null    float64
 24  worst smoothness         569 non-null    float64
 25  worst compactness        569 non-null    float64
 26  worst concavity          569 non-null    float64
 27  worst concave points     569 non-null    float64
 28  worst symmetry           569 non-null    float64
 29  worst fractal dimension  569 non-null    float64
dtypes: float64(30)
memory usage: 133.5 KB
```

In [5]:
```
# 데이터 통계량 확인
data.describe()
```

Out[5]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	c
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	56
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	

8 rows × 30 columns

In [ ]:
```
from sklearn.preprocessing import StandardScaler

# 데이터 스케일링
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data) # ndarray returned

# 스케일링된 데이터 데이터 프레임 변환 (ndarray -> Dataframe)
scaled_df = pd.DataFrame(scaled_data, columns=data.columns)
# 데이터 확인
scaled_df.head()
```

Out[ ]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	c
0	1.097064	-2.073335	1.269934	0.984375	1.568466	3.283515	2.652874	2.
1	1.829821	-0.353632	1.685955	1.908708	-0.826962	-0.487072	-0.023846	0.
2	1.579888	0.456187	1.566503	1.558884	0.942210	1.052926	1.363478	2.
3	-0.768909	0.253732	-0.592687	-0.764464	3.283553	3.402909	1.915897	1.
4	1.750297	-1.151816	1.776573	1.826229	0.280372	0.539340	1.371011	1.

5 rows × 30 columns

In [9]:
```
from sklearn.decomposition import PCA

# PCA 모델 생성 및 적용
pca = PCA(n_components=10)  # 주성분 10개로 설정
principal_components = pca.fit_transform(scaled_df)  # PCA 적용

# PCA 결과 확인
print("주성분 분석 결과 (앞 5개):\n", principal_components[:5])  # 변환된 데이터
```

```
# 설명 분산 비율 확인
explained_variance_ratio = pca.explained_variance_ratio_
```

```
주성분 분석 결과 (앞 5개):
[[ 9.19283683  1.94858307 -1.12316617  3.6337309  -1.19511012  1.41142456
   2.15936955 -0.39841327 -0.15711879 -0.87739925]
 [ 2.3878018  -3.76817174 -0.52929268  1.11826386  0.62177497  0.02865623
   0.01335865  0.24099048 -0.71190325  1.10699245]
 [ 5.73389628 -1.0751738  -0.5517476   0.91208267 -0.1770859   0.54145225
  -0.66816655  0.09737327  0.02405747  0.45428616]
 [ 7.1229532  10.27558912 -3.23278955  0.15254703 -2.9608784   3.05342179
   1.42991105  1.05957591 -1.40545131 -1.11696724]
 [ 3.93530207 -1.94807157  1.38976673  2.94063935  0.5467474  -1.22649472
  -0.93621258  0.63638139 -0.26380244  0.37769712]]
```

In [11]:
```
# 각 주성분이 설명하는 분산 비율
print("설명 분산 비율:\n", explained_variance_ratio)
```

```
설명 분산 비율:
[0.44272026 0.18971182 0.09393163 0.06602135 0.05495768 0.04024522
 0.02250734 0.01588724 0.01389649 0.01168978]
```

In [13]:
```
# 누적 설명 비율
print("누적 설명 분산 비율:\n", explained_variance_ratio.cumsum())
```

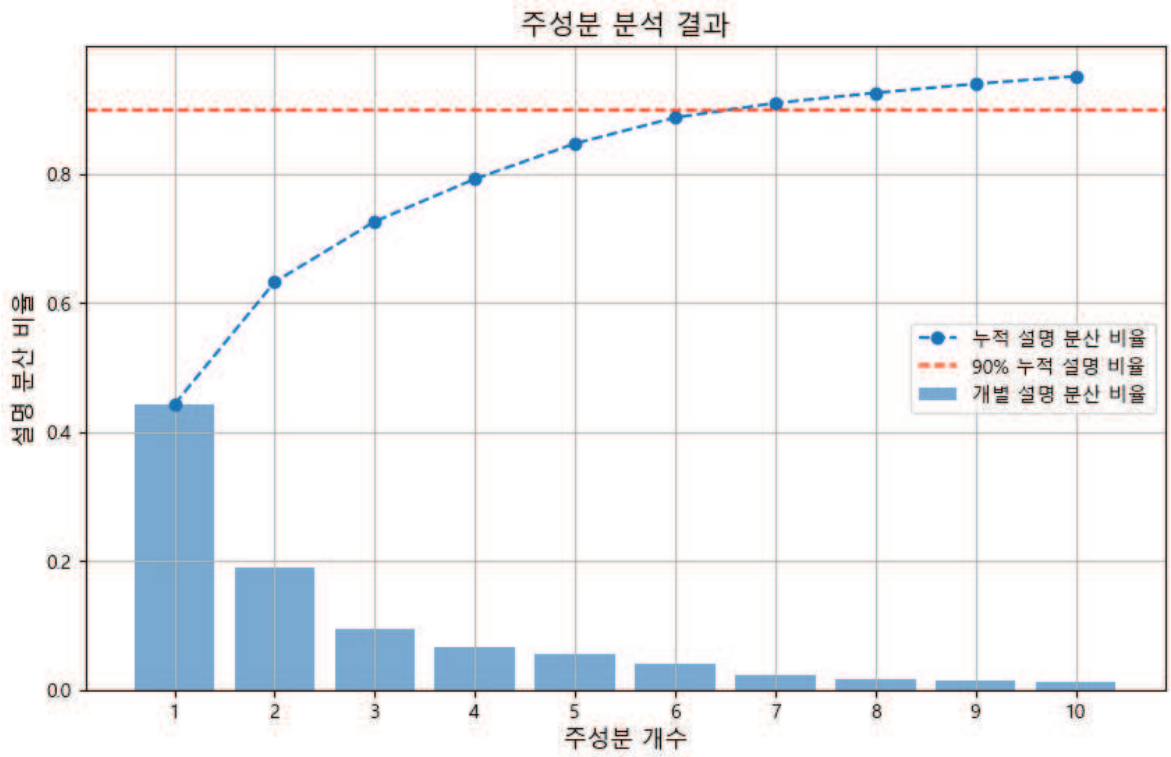
```
누적 설명 분산 비율:
[0.44272026 0.63243208 0.72636371 0.79238506 0.84734274 0.88758796
 0.9100953  0.92598254 0.93987903 0.95156881]
```

In [15]:
```
import matplotlib.pyplot as plt
from matplotlib import font_manager

# 한글 폰트 설정 (예: 'Malgun Gothic'은 윈도우에서 사용 가능)
# 맥일 경우 아래 코드 활성화
#plt.rcParams['font.family'] = 'AppleGothic'
# 윈도우일 경우 아래 코드 활성화
plt.rcParams['font.family'] = 'Malgun Gothic'

# 그래프 그리기
plt.figure(figsize=(10, 6))
plt.plot(range(1, len(explained_variance_ratio) + 1), explained_variance_ratio.c
plt.bar(range(1, len(explained_variance_ratio) + 1), explained_variance_ratio, a

# 그래프 꾸미기
plt.title('주성분 분석 결과', fontsize=14)
plt.xlabel('주성분 개수', fontsize=12)
plt.ylabel('설명 분산 비율', fontsize=12)
plt.axhline(y=0.9, color='r', linestyle='--', label='90% 누적 설명 비율')
plt.xticks(range(1, len(explained_variance_ratio) + 1))
plt.legend()
plt.grid()
plt.show()
```



In [17]:
```
# 시각화 결과에 따라 pca 재진행
pca = PCA(n_components=7)  # 주성분 7개로 설정
principal_components = pca.fit_transform(scaled_df)  # PCA 적용

# 설명 분산 비율 확인
pca.explained_variance_ratio_
```

Out[17]:
```
array([0.44272026, 0.18971182, 0.09393163, 0.06602135, 0.05495768,
       0.04024522, 0.02250734])
```