1.The sizeof result for these new types.

_BitInt : 4bytes

Unsigned _BitInt : 8bytes

```c
#include <stdio.h>

#define BITINT_MAXWIDTH 64 // Example maximum width for _BitInt(N)

typedef _BitInt(32) BitInt32;
typedef unsigned _BitInt(64) UnsignedBitInt64;

int main() {
    printf("Size of _BitInt(32): %zu\n", sizeof(BitInt32));
    printf("Size of unsigned _BitInt(64): %zu\n", sizeof(UnsignedBitInt64));
    return 0;
}
```

```
Size of _BitInt(32): 4
Size of unsigned _BitInt(64): 8
```

2.What will happen if you assign a value to an integer which is over the given size?

會 overflow(以二補數溢出的方式)

```c
#include <stdio.h>

typedef _BitInt(8) BitInt8;

int main() {
    BitInt8 num = 300;
    printf("Value of num: %d\n", num);
    return 0;
}
```

```
Value of num: -44
```

3.Is it possible to do arithmetic operations between two operands with different bits?

可以使用不同位元寬度的算術運算,結果是較寬的位元寬度

```c
#include <stdio.h>

typedef _BitInt(8) BitInt8;
typedef _BitInt(16) BitInt16;

int main() {
    BitInt8 num1 = 50;
    BitInt16 num2 = 1000;
    BitInt16 result = num1 + num2;
    printf("Result: %d\n", result);
    return 0;
}
```

```
Result: 1050
```

4.Is it possible to do arithmetic operations between _BitInt(N) and unsigned _BitInt(N)?If yes, the result is _BitInt(N) or unsigned _BitInt(N)?

可以,且結果為有號數

```c
#include <stdio.h>

typedef _BitInt(8) BitInt8;
typedef unsigned _BitInt(8) UnsignedBitInt8;

int main() {
    BitInt8 num1 = -10;
    UnsignedBitInt8 num2 = 20;
    BitInt8 result = num1 - num2;
    printf("Result: %d\n", result);
    return 0;
}
```

```
Result: -30
```

5.What will happen if N is greater than BITINT_MAXWIDTH?

在編譯時就會收到錯誤訊息,無法這樣定義

```c
#include <stdio.h>

#define BITINT_MAXWIDTH 64 // Maximum width for _BitInt(N)

typedef _BitInt(128) BitInt128; // Attempting to define a _BitInt with width greater

int main() {
    printf("Size of _BitInt(128): %zu\n", sizeof(BitInt128));
    return 0;
}
```

```
error: width of '_BitInt' exceeds BITINT_MAXWIDTH (which is 64)
```