1. Typedef for Integer:

```c
#include <stdio.h>

typedef int MyInt;

int main() {
    MyInt num = 42;
    printf("Value of num: %d\n", num);
    return 0;
}
```

2. Typedef for Character

```c
#include <stdio.h>

typedef char MyChar;

int main() {
    MyChar ch = 'A';
    printf("Character: %c\n", ch);
    return 0;
}
```

3. Typedef for Floating-Point:

```c
#include <stdio.h>

typedef float MyFloat;

int main() {
    MyFloat pi = 3.14159265359;
    printf("Value of pi: %.2f\n", pi);
```

```
    return 0;
}
```

4. Typedef for Array:

```
#include <stdio.h>

typedef int IntArray[5];

int main() {
    IntArray arr = {1, 2, 3, 4, 5};
    printf("Array elements: %d %d %d %d %d\n", arr[0], arr[1], arr[2],
arr[3], arr[4]);
    return 0;
}
```

5. Typedef for Pointer:

```
#include <stdio.h>

typedef int* IntPtr;

int main() {
    int num = 42;
    IntPtr p = &num;
    printf("Value through pointer: %d\n", *p);
    return 0;
}
```

6. Typedef for Structure:

```c
#include <stdio.h>

typedef struct {
    int x;
    int y;
} Point;

int main() {
    Point p1 = {1, 2};
    printf("Point coordinates: (%d, %d)\n", p1.x, p1.y);
    return 0;
}
```

7. Typedef for Enumeration:

```c
#include <stdio.h>

typedef enum {
    MONDAY,
    TUESDAY,
    WEDNESDAY,
    THURSDAY,
    FRIDAY
} Weekday;

int main() {
    Weekday today = MONDAY;
    printf("Today is day %d\n", today);
    return 0;
}
```

8. Typedef for Function Pointer:

```c
#include <stdio.h>

typedef int (*MathFunc)(int, int);

int add(int a, int b) {
    return a + b;
}

int main() {
    MathFunc func = add;
    int result = func(5, 3);
    printf("Result of addition: %d\n", result);
    return 0;
}
```

9. Typedef for Complex Data Type:

```c
#include <stdio.h>

typedef struct {
    char name[20];
    int age;
} Person;

int main() {
    Person john;
    strcpy(john.name, "John");
    john.age = 30;
    printf("Name: %s, Age: %d\n", john.name, john.age);
    return 0;
}
```

10. Typedef for Pointer to Function:

```c
#include <stdio.h>

typedef int (*ArithmeticFunc)(int, int);

int add(int a, int b) {
    return a + b;
}

int subtract(int a, int b) {
    return a - b;
}

int main() {
    ArithmeticFunc func;
    func = add;
    printf("Addition: %d\n", func(5, 3));
    func = subtract;
    printf("Subtraction: %d\n", func(5, 3));
    return 0;
}
```

11. Typedef for Pointer to Array:

```c
#include <stdio.h>

typedef int (*IntArrayPtr)[5];

int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    IntArrayPtr ptr = &arr;
```

```
    printf("First element: %d\n", (*ptr)[0]);
    return 0;
}
```

12. Typedef for Character Pointer:

```
#include <stdio.h>

typedef char* String;

int main() {
    String name = "Alice";
    printf("Name: %s\n", name);
    return 0;
}
```

13. Typedef for Function Returning Pointer:

```
#include <stdio.h>

typedef int* IntPtrFunc(int);

int* doubleValue(int x) {
    int* result = malloc(sizeof(int));
    *result = x * 2;
    return result;
}

int main() {
```

```
    IntPtrFunc* func = doubleValue;
    int* result = func(5);
    printf("Doubled value: %d\n", *result);
    free(result);
    return 0;
}
```

14. Typedef for Multi-dimensional Array:

```
#include <stdio.h>

typedef int Matrix[3][3];

int main() {
    Matrix identity = {{1, 0, 0}, {0, 1, 0}, {0, 0, 1}};
    printf("Identity matrix element (1,1): %d\n", identity[0][0]);
    return 0;
}
```

15. Typedef for Unsigned Data Type:

```
#include <stdio.h>

typedef unsigned int MyUnsignedInt;

int main() {
    MyUnsignedInt num = 42;
    printf("Value of unsigned integer: %u\n", num);
    return 0;
}
```

16. Typedef for Signed and Unsigned Types:

```c
#include <stdio.h>

typedef signed char MySignedChar;
typedef unsigned char MyUnsignedChar;

int main() {
    MySignedChar a = -1;
    MyUnsignedChar b = 255;
    printf("Signed Char: %d, Unsigned Char: %u\n", a, b);
    return 0;
}
```

17. Typedef for Short and Long Types:

```c
#include <stdio.h>

typedef short MyShort;
typedef long MyLong;

int main() {
    MyShort s = 32767;
    MyLong l = 2147483647;
    printf("Short: %hd, Long: %ld\n", s, l);
    return 0;
}
```

18. Typedef for Complex Structures:

```c
#include <stdio.h>

typedef struct {
    int x;
    int y;
} Point;

typedef struct {
    char name[20];
    int age;
} Person;

int main() {
    Point p = {10, 20};
    Person person = {"Alice", 30};
    printf("Point: (%d, %d), Person: %s, %d years old\n", p.x, p.y,
person.name, person.age);
    return 0;
}
```

19. Typedef for Enum and Integer:

```c
#include <stdio.h>

typedef enum {
    RED,
    GREEN,
    BLUE
} Color;

typedef int IntColor;
```

```c
int main() {
    Color myColor = GREEN;
    IntColor myIntColor = 42;
    printf("Enum Color: %d, Integer Color: %d\n", myColor, myIntColor);
    return 0;
}
```

20. Typedef for Pointer to Function Returning Pointer:

```c
#include <stdio.h>

typedef int* (*FuncReturningIntPtr)();

int* createIntArray() {
    int* arr = malloc(5 * sizeof(int));
    for (int i = 0; i < 5; i++) {
        arr[i] = i;
    }
    return arr;
}

int main() {
    FuncReturningIntPtr func = createIntArray;
    int* arr = func();
    printf("Array elements: %d %d %d %d %d\n", arr[0], arr[1], arr[2],
arr[3], arr[4]);
    free(arr);
    return 0;
}
```

++++++++++++++++++++++++++++++++Happy Learning ++++++++++++++++++++++++++++++