```
1. Allocate an int
2. Allocate two ints
3. Allocate three ints
4. Allocate one char
5. Allocate space for an 80-element int array
6. Quit
Choose a menu option: 1

---Test Case 1---
Address of int A: 0000000000A96A40
Address of int B: 0000000000A96A40

1. Allocate an int
2. Allocate two ints
3. Allocate three ints
4. Allocate one char
5. Allocate space for an 80-element int array
6. Quit
Choose a menu option: 2

---Test Case 2---
Address of int A: 0000000000A91430
Address of int B: 0000000000A91448
Verifying Results...
Size of overhead + larger of (the size of an integer; the minimum block size): 24 bytes
Address B - Address A: 24 bytes

1. Allocate an int
```

```
2. Allocate two ints
3. Allocate three ints
4. Allocate one char
5. Allocate space for an 80-element int array
6. Quit
Choose a menu option: 3

---Test Case 3---
Address of int A: 0000000000A96E30
Address of int B: 0000000000A96E48
Address of int C: 0000000000A96E60
After freeing int B...
Address of array of 2 double values: 0000000000A96E78
Address of int D (should be the int B): 0000000000A96E48

1. Allocate an int
2. Allocate two ints
3. Allocate three ints
4. Allocate one char
5. Allocate space for an 80-element int array
6. Quit
Choose a menu option: 4

---Test Case 4---
Address of char A: 0000000000A97220
Address of int B: 0000000000A97238
Size of overhead + larger of (the size of an integer; the minimum block size): 24

1. Allocate an int
```

```
2. Allocate two ints
3. Allocate three ints
4. Allocate one char
5. Allocate space for an 80-element int array
6. Quit
Choose a menu option: 5

---Test Case 5---
Address of array: 0000000000A97610
Address of int A: 0000000000A97760
Address of int value: 0000000000A97760
Value of int A: 0
Difference betwween array and int A: 336
After freeing array...
Address of int value: 0000000000A97760
Value of int A: 0

1. Allocate an int
2. Allocate two ints
3. Allocate three ints
4. Allocate one char
5. Allocate space for an 80-element int array
6. Quit
Choose a menu option:
```