

Referencing Nonexistent Variables

- Forgetting that local variables disappear when a function returns

```
int *foo () {  
    int val;  
  
    return &v  
}
```

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Freeing Blocks Multiple Times

■ Nasty!

```
x = malloc(N*sizeof(int));
```

<manipulate x>

```
free(x);
```

```
y = malloc(M*
```

<manipulate y>

```
free(x);
```

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Referencing Freed Blocks

■ Evil!

```
x = malloc(N*sizeof(int));
```

```
    <manipulate x>
```

```
free(x);
```

```
    ...
```

```
y = malloc(M*
```

```
for (i=0; i<M
```

```
    y[i] = x[i];
```

Failing to Free Blocks (Memory Leaks)

- Slow, long-term killer!

```
foo() {  
    int *x = malloc(N*sizeof(int));  
    ...  
    return;  
}
```

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Failing to Free Blocks (Memory Leaks)

- Freeing only part of a data structure

```
struct list {  
    int val;  
    struct list *next;  
};  
  
foo() {  
    struct list *head = malloc(sizeof(struct list));  
    head->val = 0;  
    head->next = NULL;  
    <create and manipulate the rest of the list>  
    ...  
    free(head);  
    return;  
}
```

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro