Name: | | Login: | | IN-CLASS EXERCISE

Address syntax 2

For this exercise, assume: `sizeof(int)==4 && sizeof(char)==1 && sizeof(void*)==8`

SOLUTION

Initializing new variables

```
// c1 is a char initialized to 55 ('7') with an integer literal
char c1 = 55;
```

sizeof(c2) == 1

```
// c2 is a char initialized to 53 ('5') with an integer literal
char c2 = 53;
```

sizeof(c2) == 1

```
// s1 is the address of the first char in a string stored in the data
// segment: "75"
char *s1 = "75";
```

sizeof(s1) == 8

```
// s2 is an array of char (a string) stored on the stack and initialized
// to "75" using a string literal.
char s2[] = "75";
```

sizeof(s2) == 3

```
// s3 is an array of char (a string) stored on the stack and initialized
// to "75" using an array initializer containing character literals.
char s3[] = {'7', '5', '\0'};
```

sizeof(s3) == 3

```
// s4 is an array of char (a string) stored on the stack and initialized
// to "75" using an array initializer containing integer literals.
char s4[] = {55, 53, 0};
```

sizeof(s4) == 3

```
// s5 is the address of c1.
char* s5 = &c1;
```

Output: 8

```
// a_s5 is the address of s5.
char** a_s5 = &st;
```

Output: 8

Copy from page 1

```
// s4 is an array of char (a string) stored on the stack and initialized
// to "75" using an array initializer containing integer literals.
char s4[] = {55, 53, 0};
```

```
// s5 is the address of c1.
char* s5 = &c1;
```

```
// a_s5 is the address of s5.
char** a_s5 = &st;
```

Using addresses in expressions

```
// Print s4 using ordinary C (i.e., not mintf).
printf(s4);
```

Output: 75

```
// Print s5 without using the variable name c1.  Use s5, *, and ordinary C.
fputc(*s5, stdout)
```

Output: 7

```
// Print s5 without using the variable name c1.  Use s5, […] and ordinary C.
fputc(s5[0], stdout)
```

Output: 7

```
// Print s5 without using the variable name c1.  Use a_s5, *, and ordinary C.
fputc(**a_s5, stdout)
```

Output: 7

Assignments

```
// Store '?' in c1 without using the variable name c1.  Use s5 and *.
*s5 = '?';
```

sizeof(s5) == 8

```
// Store '@' in c1 without using the variable name c1.  Use s5 and […].
*s5 = '@';
```

sizeof(s5) == 8

```
// s5 gets the address of c2.
*s5 = &c2;
```

sizeof(s5) == 8

```
// s5 gets the address of the last character in s4.
*s5 = &(s4[2]);
```

sizeof(s5) == 8