Universidad del Valle

Facultad de Ingeniería
Escuela de Ingeniería de Sistemas y Computación

Seguridad en el Desarrollo de Aplicaciones

Maestría en Computación para el Desarrollo de Aplicaciones Inteligentes [CODING]

Maestría Virtual en Ingeniería de Sistemas y Computación

Q1 (Control Flow): What output does the following code produce?

```c
#include<stdio.h>

int main(void) {
  int x, y;
  int i = 3;
  x = y = 0;
  while (i >= 0) {
    x += 1;
    i--;
  }
  do {
    y += 1;
    i++;
  }
  while (i <= 3);

  printf("%d\n", (x - y));
}
```

Answers

- -2
- -1
- 0
- 1

Q2 (Pointers): What output does the following code produce?

```
#include<stdio.h>

int main(void) {
  int x, y;

  int ary[4] = { 11, 22, 33, 44 };
  int *ptr;

  ptr = ary;
  ptr++;

  printf("%d %p\n", *ptr, ptr);
}
```

Answers

- 11 0x7ffc3ae795e0
- 22 0x7ffc3ae795e4
- 33 0x7ffc3ae795e8
- 44 0x7ffc3ae795ec

Q3 (Pointers): What does the following program print when executed with arguments: one two three?

```
#include <stdio.h>

int main(int argc, char *argv[]) {
  for(char *p; (p = *++argv);) {
    for (; *p; ++p)
      putchar(*p);
  }
}
```

Answers

- three
- onetwothree
- NULL
- twothree

Q4 (arguments are passed by value): What does the following function print?

```
#include <stdio.h>
```


```
void swap(int x, int y) {
  int tmp = x;
  x = y;
  y = tmp;
}

int main(void) {
  int a = 33;
  int b = 55;

  swap(a,b);
  printf("%d %d\n",a,b);
}
```

Answers

- 33 55
- 55 33
- Error
- NULL

Q5 (arguments are passed by value): What does the following function print?

```
#include <stdio.h>

void swap(int* x, int* y) {
  int tmp = *x;
  *x = *y;
  *y = tmp;
}

int main(void) {
  int a = 33;
  int b = 55;

  swap(&a,&b);
  printf("%d %d\n",a,b);
}
```

Answers

- 33 55
- 55 33
- Error
- NULL

Q6 (Assignments): What does the following program print?

```
#include<stdio.h>
```


```c
int main() {
  int x, y, z;
  x = y = z = 1;
  x += (y += z);
  printf("%d %d %d\n", x, y, z);
}
```

Answers

- 1 1 1
- 1 2 3
- 3 3 3
- 3 2 1
- Compiling error

Q7 (Assignments): What does the following program print?

```c
#include<stdio.h>

int main() {
  int x, y, z;
  x = y = z = 1;
  (x += y) += z;
  printf("%d %d %d\n", x, y, z);
}
```

Answers

- 1 1 1
- 1 2 3
- 3 3 3
- 3 2 1
- Compile-time error

Q8 (Operators): What does the following program print?

```c
#include<stdio.h>

void initArray() {
  int array[3];
  int i = 3, x = 3;

  while (--i && (array[i] = --x))
    printf("%d ", array[i]);
}

int main() {
```


```
  initArray();
}
```

Answers

- 3 3
- 2 1
- 3 2
- 2 3
- 1 1

Q9 (Strings): what does the following program print?

```
int main() {

	char *t;
	const char *str = "item1 item2 item3 item4 item5";

	char *tmp = (char *)malloc(strlen(str) + 1);
	if (tmp == NULL) {
		exit(1);
	}
	strcpy(tmp, str);
	t = strtok(tmp, " ");
	printf("%s,", t);

	while (t = strtok(0, " ")) {
		printf("%s,", t);
	}

	free(tmp);
	tmp = NULL;

}
```

Answers

- item1 item2 item3 item4 item5
- item1 item2 item3 item4
- item1, item2, item3, item4, item5
- item1, item2, item3, item4, item5,
- Run-time error

Q10 (Strings): what does the following program print?

```
#include<stdio.h>

int main() {
  char arr[] = "abcc";

  *(arr + 3) = 'd';
  puts(arr);
```


```
  return 0;
}
```

Answers

- abcc
- Run-time error
- abcd
- abccd

Q11 (Strings): what does the following program print?

```
#include<stdio.h>

int main() {
  char *str = "abcc";

  *(str + 3) = 'd';
  puts(str);

  return 0;
}
```

Answers

- abcc
- Run-time error
- abcd
- abccd

Q12 (Functions): what does the following program print?

```
#include<stdio.h>

#define SIZE 3

int sum(int a1[], int a2[]) {

  int sum = 0;

  for (int i = 0; i < SIZE; i++) {
    a1[i] += a2[i];
    sum += a1[i];
  }

  return sum;
}
```

```
int main() {

  int array1[SIZE] = {1,2,3};
  int array2[SIZE] = {1,2,3};

  int x = sum(array1, array2);

  for (int i = 0; i < SIZE; i++) {
    printf("%d + %d", array1[i], array2[i]);
    if (i < SIZE - 1) printf(" + ");
  }
  printf(" = %d\n", x);

}
```

Answers

- 6
- 12
- 18
- 24

Q13 (Structures): what does the following program print?

```
#include<stdio.h>

struct Date
{
  int day;
  int month;
  int year;
};

int main()
{
  struct Date d = { 1, 1, 2020 };

  struct Date *d2;
  d2 = &d;

  printf("%d/%d/%d\n", d2->day, d2->month, d2->year);
  return 0;
}
```

Answers

- 2020/1/1
- 1/2020/1
- 1/1/2020
- 1/1//1

Q14 (Structures): what does the following program print?

```
#include<stdio.h>

struct {
  char s[3];
  int i;
} x;

int main()
{
  printf("%ld\t", (char *)&x.i - x.s);
  printf("%lu\n", sizeof(x) - sizeof(x.s) - sizeof(x.i));
}
```

Answers

- 4 1
- 3 1
- 6 2
- 4 2

Q15 (Inputs, Outputs, and Strings): What is the behaviour of the following program one the input line one\nline two\n?

```
#include<stdio.h>
#include<string.h>

int main()
{
  char s[10];
 fgets(s, 10, stdin);
 printf("%lu", strlen(s));
}
```

Answers

- It reads a line, until reaching \n, EOF, or the limit (10-1 chars). For the given input, the string read is “line one\n” (total 9 characters).
- It reads a word, until reaching whitespace or 9 characters. For the given input, the string read s is: “line”.

- It reads exactly 10 characters of any kind (including spaces or terminators), stopping only at EOF. s[] will contain “line one\nl” (ending with the first ‘l’ character of line two), with no \0 terminator.

Q16 (Inputs, Outputs, and Strings): What is the behaviour of the following program one the input line one\nline two\n?

```
#include<stdio.h>
#include<string.h>

int main()
{
  char s[10];
  scanf("%9s", s);
  printf("%zu", strlen(s));

}
```

Answers

- It reads a line, until reaching \n, EOF, or the limit (10-1 chars). For the given input, the string read is “line one\n” (total 9 characters).
- It reads a word, until reaching whitespace or 9 characters. For the given input, the string read s is: “line”.
- It reads exactly 10 characters of any kind (including spaces or terminators), stopping only at EOF. s[] will contain “line one\nl” (ending with the first ‘l’ character of line two), with no \0 terminator.

Q17 (Inputs, Outputs, and Strings): What is the behaviour of the following program on the input line one\nline two\n?

```
#include<stdio.h>
#include<string.h>

int main()
{
  char s[10];
  scanf("%10c", s);
  printf("%zu", strlen(s));
}
```

Answers

- It reads a line, until reaching \n, EOF, or the limit (10-1 chars). For the given input, the string read is “line one\n” (total 9 characters).
- It reads a word, until reaching whitespace or 9 characters. For the given input, the string read s is: “line”.
- It reads exactly 10 characters of any kind (including spaces or terminators), stopping only at EOF. s[] will contain “line one\nl” (ending with the first ‘l’ character of line two), with no \0 terminator.

Q18 (Strings): What can you tell about s1 and s2 in the following program?

```
int main()
{
    char s1[5] = "Hello";
    char s2[5] = "cStr";

    ...
    return 0;
}
```

Answers

- s1 ends with '\0' but s2 does not.
- s2 ends with '\0' but s1 does not.
- Both of s1 and s2 ends with '\0'.
- None of s1 and s2 ends with '\0'.

Q19 (heap vs stack): Does the following program print 33?

```
#include <stdlib.h>
#include <stdio.h>

int* f() {
  int* p = malloc(sizeof(int));
  (*p) = 33;
  return p;
}

int main(void) {
  int* p = f();
  printf("%i\n",(*p));
  return 0;
}
```

Answers

- Yes
- No

Q20 (heap vs stack): Does the following program print 33?

```
#include <stdlib.h>
#include <stdio.h>
```


```c
int* f() {
  int p = 33;
  return &p;
}

int main(void) {
  int* p = f();
  printf("valor de p: %i\n",(*p));

  return 0;
}
```

Answers

- Yes
- No