C LANG TEST-5 (LOOP INTERRUPTIONS)

Total points 50/50

STUDENT NAME *

VIVA

✓ 1. **Which statement is used to terminate a loop immediately in C?** * 1/1

- ○ A) exit
- ○ B) stop
- ◉ C) break ✓
- ○ D) continue

✓ 2. **Which statement skips the rest of the statements in the current loop iteration?** *1/1

- ○ A) break
- ◉ B) continue ✓
- ○ C) goto
- ○ D) exit

✓ 3. **The break statement can be used inside:** * 1/1

- ○ A) for loop
- ○ B) while loop
- ○ C) switch statement
- ◉ D) All of the above ✓

✓ 4. **The continue statement can be used inside:** * 1/1

- ○ A) switch
- ◉ B) for loop ✓
- ○ C) function
- ○ D) structure

✓ 5. **Which statement transfers control to a labeled statement?** * 1/1

- ○ A) exit
- ◉ B) goto ✓
- ○ C) break
- ○ D) continue

✓ 6. **What happens when break is executed in a loop?** * 1/1

- (●) A) Exits the loop ✓
- ( ) B) Skips next iteration
- ( ) C) Repeats the loop
- ( ) D) Restarts the loop

✓ 7. **What happens when continue is executed in a loop?** * 1/1

- ( ) A) Ends the loop
- ( ) B) Restarts the loop
- (●) C) Skips the remaining statements and continues next iteration ✓
- ( ) D) Stops the program

✓ 8. **Which of the following cannot be used with switch statement?** * 1/1

- ( ) A) break
- (●) B) continue ✓
- ( ) C) goto
- ( ) D) return

✓ 9. **What is the output?** * 1/1

```
for(int i=1;i<=5;i++){
  if(i==3) break;
  printf("%d ", i);
}
```

- ○ A) 1 2 3 4 5
- ◉ B) 1 2 ✓
- ○ C) 3 4 5
- ○ D) 1

✓ 10. **What is the output?** * 1/1

```
for(int i=1;i<=5;i++){
  if(i==3) continue;
  printf("%d ", i);
}
```

- ○ A) 1 2 3 4 5
- ◉ B) 1 2 4 5 ✓
- ○ C) 3 4 5
- ○ D) 1 3 5

✓ 11. **The keyword used to exit from the entire program is:** * 1/1

- A) break
- B) continue
- (●) C) exit() ✓
- D) goto

✓ 12. **Which header file is required for exit() function?** * 1/1

- A) stdio.h
- (●) B) stdlib.h ✓
- C) conio.h
- D) process.h

✓ 13. **Which statement is used to jump from one part of code to another?** * 1/1

- A) continue
- B) break
- (●) C) goto ✓
- D) skip

✓ 14. **Using break outside a loop or switch results in:** * 1/1

- (●) A) Compile-time error ✓
- ( ) B) Logical error
- ( ) C) Infinite loop
- ( ) D) No error

✓ 15. **Using continue outside a loop results in:** * 1/1

- (●) A) Compile-time error ✓
- ( ) B) Logical error
- ( ) C) Infinite loop
- ( ) D) Runtime error

✓ 16. **What is the output?** * 1/1

```
for(int i=0;i<3;i++){
  printf("A ");
  continue;
  printf("B ");
}
```

- (●) A) A A A ✓
- ( ) B) A B A B A B
- ( ) C) B B B
- ( ) D) Error

✓ 17. **Which of these is a jump statement in C?** * 1/1

- ○ A) break
- ○ B) continue
- ○ C) goto
- ◉ D) All of these ✓

✓ 18. **The goto statement is also known as:** * 1/1

- ○ A) Loop breaker
- ◉ B) Unconditional jump ✓
- ○ C) Loop creator
- ○ D) Conditional branch

✓ 19. **The label used in a goto statement must end with:** * 1/1

- ○ A) Semicolon
- ◉ B) Colon ✓
- ○ C) Comma
- ○ D) Period

✓ 20. **What happens if label in goto is missing?** * 1/1

- ○ A) Logical error
- ◉ B) Compilation error ✓
- ○ C) Runtime error
- ○ D) No effect

✓ 21. **What happens when break is used in nested loops?** * 1/1

- ○ A) Terminates all loops
- ◉ B) Terminates only the innermost loop ✓
- ○ C) Terminates outermost loop
- ○ D) None

✓ 22. **What is the output?** * 1/1

```
for(int i=1;i<=3;i++){
  for(int j=1;j<=3;j++){
    if(j==2) break;
    printf("%d%d ", i,j);
  }
}
```

- ○ A) 11 12 13 21 22 23 31 32 33
- ◉ B) 11 21 31 ✓
- ○ C) 12 22 32
- ○ D) None

✓ 23. . **What is the output?** * 1/1

```
for(int i=1;i<=3;i++){
 if(i==2) continue;
 printf("%d ", i);
}
```

- ○ A) 1 2 3
- ◉ B) 1 3 ✓
- ○ C) 2 3
- ○ D) 1 2

✓ 24. **What does this code do?** * 1/1

```
while(1){
 if(x>5) break;
}
```

- ○ A) Infinite loop
- ◉ B) Terminates when x>5 ✓
- ○ C) Compile error
- ○ D) None

✓ 25. **What is the output?** * 1/1

```
int i=0;
while(i<5){
 i++;
 if(i==3) break;
 printf("%d ", i);
}
```

- (●) A) 1 2 ✓
- ( ) B) 1 2 3
- ( ) C) 3 4 5
- ( ) D) 0 1 2

✓ 26. **The continue statement in a for loop transfers control to:** * 1/1

- ( ) A) Start of loop
- ( ) B) Condition check part
- (●) C) Increment/decrement part ✓
- ( ) D) Loop body

✓ 27. **The continue statement in a while loop transfers control to:** * 1/1

- (●) A) Beginning of loop ✓
- ( ) B) Next iteration check
- ( ) C) End of loop
- ( ) D) Exit

✓ 28. **Which is true about goto statement?** * 1/1

- A) It can jump backward
- B) It can jump forward
- C) It can jump both ways
- D) Only within same function ✓

✓ 29. **Which is the best alternative to goto for structured programming?** * 1/1

- A) switch
- B) loops ✓
- C) functions
- D) recursion

✓ 30. **The statement used to exit from current loop iteration is:** * 1/1

- A) exit
- B) continue ✓
- C) stop
- D) break

✓ 31. **What is the output?** * 1/1

```
int i=0;
do{
 i++;
 if(i==2) continue;
 printf("%d ", i);
}while(i<4);
```

- ○ A) 1 2 3 4
- ◉ B) 1 3 4 ✓
- ○ C) 2 3 4
- ○ D) 1 2 3

✓ 32. **The break statement causes control to pass to:** * 1/1

- ○ A) The start of next iteration
- ◉ B) The next statement after loop ✓
- ○ C) The top of loop
- ○ D) The previous loop

✓ 33. **What happens if you use goto to jump inside a loop from outside?** * 1/1

- ○ A) Valid
- ○ B) Undefined behavior
- ○ C) Infinite loop
- ◉ D) Compile error ✓

✓ 34. **What is the output?** * 1/1

```
for(int i=0;i<3;i++){
  printf("%d ", i);
  if(i==1) break;
}
```

- (●) A) 0 1 ✓
- ( ) B) 0 1 2
- ( ) C) 1 2
- ( ) D) 0

✓ 35. **break cannot be used inside:** * 1/1

- ( ) A) for
- ( ) B) while
- ( ) C) do-while
- (●) D) if ✓

✓ 36. **What is the effect of exit(0) in a program?** * 1/1

- ( ) A) Ends current function
- (●) B) Ends program successfully ✓
- ( ) C) Ends loop only
- ( ) D) Restarts program

✓ 37. **The exit() function terminates:** * 1/1

- ○ A) The innermost loop
- ○ B) Current function
- ◉ C) Entire program ✓
- ○ D) The compiler

✓ 38. **What is the output?** * 1/1

```
int i;
for(i=1;i<=5;i++){
  if(i==4) goto end;
  printf("%d ", i);
}
```

- ◉ A) 1 2 3 End ✓
- ○ B) 1 2 3 4 5 End
- ○ C) End only
- ○ D) Error

✓ 39. **What is the output?** * 1/1

```
int i=1;
while(i<=3){
 if(i==2) continue;
 printf("%d ", i);
 i++;
}
```

- ○ A) 1 2 3
- ◉ B) Infinite loop ✓
- ○ C) 1 3
- ○ D) 2 3

✓ 40. **Why does the above loop become infinite?** * 1/1

- ○ A) Missing semicolon
- ◉ B) continue skips increment ✓
- ○ C) Wrong syntax
- ○ D) Break not used

✓ 41. **What is the output?** * 1/1

```
int i=0;
while(i<4){
 printf("%d ", i);
 if(i==2) break;
 i++;
}
```

- ○ A) 0 1 2 3
- ◉ B) 0 1 2 ✓
- ○ C) 0 1
- ○ D) None

✓ 42. **Can goto jump into another function?** * 1/1

- ○ A) Yes
- ◉ B) No ✓
- ○ C) Only in recursion
- ○ D) Depends on compiler

✓ 43. **. The label in goto statement must be:** * 1/1

- ○ A) Integer
- ◉ B) Identifier ✓
- ○ C) Keyword
- ○ D) String

✓ 44. **What happens when continue executes in nested loops?** * 1/1

- ○ A) Skips entire loops
- ◉ B) Skips current iteration of inner loop ✓
- ○ C) Skips outer loop
- ○ D) None

✓ 45. **Can break and continue be used together in same loop?** * 1/1

- ◉ A) Yes ✓
- ○ B) No

✓ 46. **What is the output?** * 1/1

```
for(int i=1;i<=5;i++){
  if(i%2==0) continue;
  printf("%d ", i);
}
```

- ○ A) 1 2 3 4 5
- ○ B) 2 4
- ◉ C) 1 3 5 ✓
- ○ D) None

✓ 47. **Which statement can replace break in logic but not syntax?** * 1/1

- (●) A) goto ✓
- ( ) B) continue
- ( ) C) if
- ( ) D) exit

✓ 48. **What will be printed?** * 1/1

```
for(int i=0;i<5;i++){
  if(i==3) exit(0);
  printf("%d ", i);
}
printf("Done");
```

- ( ) A) 0 1 2 3 Done
- ( ) B) 0 1 2 Done
- (●) C) 0 1 2 ✓
- ( ) D) None

✓ 49. **What is a potential problem with goto?** * 1/1

- ( ) A) Compilation error
- (●) B) Makes code unstructured and hard to debug ✓
- ( ) C) Makes loops faster
- ( ) D) None

✓ 50. **Which of the following statements about loop interruptions is true?** * 1/1

- ○ A) break exits loop
- ○ B) continue skips iteration
- ○ C) goto jumps to label
- ◉ D) All of the above ✓

This content is neither created nor endorsed by Google. - Contact form owner - Terms of Service - Privacy Policy

Does this form look suspicious? Report

Google Forms