Podstawy Programowania – zaliczenie

Wykład

1. Jaka jest podstawowa różnica między kompilatorem a interpreterem?

Interpreter - analizuje instrukcje programu źródłowego a przeanalizowane fragmenty (zazwyczaj jedna instrukcja) są wykonywane. Wykonanie powtórnie tego samego fragmentu wymaga powtórnej analizy.

Kompilator - to program tłumaczący program w języku wysokiego poziomu, który tworzy programy wynikowe, uruchamialne dopiero po zakończeniu tłumaczenia.

Różnica polega na tym, że Interpreter tłumaczy napisany kod we fragmentach, które są wykonywane, a Kompilator tłumaczy cały program, który można dopiero wykonać po całkowitym tłumaczeniu.

2. Liczba "a" przyjmuje wartości całkowite z zakresu od -25000 do 127.

a) Zdefiniuj typ zmiennej
b) Zdefiniuj stałą "r" równą 1,2
a) int a;
b) const float/double r=1,2;

3.

a) Zamień na liczbę dziesiętną 0xAA
b) Przedstaw w kodzie NKB (Naturalny Kod Binarny) liczbę 192
a) 170
b) 1100 0000

4. Podaj wynik działania kodu:

```
unsigned int a = -100;
int b = 100;
if (a<b) printf("aaa");
else printf("bbb");
```

Wynik:

NIEOKREŚLONY

5. Zapisz pustą pętlę główną w standardzie ANSI C.

```
int main()
{
return 0;
}
```

6. Znajdź błędy w kodzie:

```
int main(){
int a,b,c,d,e,f;g,h,i,j;
a:=0;
b+=2;
}
```

BŁĘDY:

```
int main(){
int a,b,c,d,e,f;g,h,i,j;
a:=0;
b+=2;

return 0;
}
```

Średnik między "f" i "g", ~~a:=0~~, brak "return 0"

7. Jaką wartość wyświetli program:

```
int a;
int main(){
return a;
}
```

Wyświetli 0, ponieważ a to zmienna globalna.

8. Jaką wartość przyjmie a?

```
a = 1 <<5;
```

32

9. Co pojawi się na ekranie?

```
a=2, b=4;
if (a%=b) printf("xxx");
else printf("yyy");
```

xxx

a%=b →a=a%b → a=2%4 → a=2 → if (2)

10. Napisz kawałem kodu podstawiający za "a" wartość 44 i drukujący na ekranie wartość "a" w formacie 0044.00.

```
int a=44;
printf("a=%07.2d", a);
```

07 → ilość znaków zapisanych w tym formacie; .2→ ilość znaków po kropce

11. Podaj wartość x i y dla których będą prawdziwe wyrażenia

a) if (x && y)
b) if (x & y)
a) jeśli x i y != 0 - wyrażenia logiczne
b) iloczyn bitowy np. x=5 y=1; lub x=y

12. Co zostanie wyświetlone na ekranie?

```
liczba =1;
switch (liczba)
{
case 0: printf("\nliczba 0");
case 1: printf("\nliczba 1");
case 2: printf("\nliczba 2");
default:printf("\nLiczba >2);
}
```

Wynik:

liczba 1
liczba 2
Liczba >2

13. Jaka będzie wydrukowana wartość a?

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

void func1();

int main()
{
int a=50;
func1();
printf("a=%d", a);
getche();
return 0;
}

void func1()
{
int a=100;
}
```

Wynik:
a=50;

14.Co zostanie wydrukowane na ekranie?

```
unsigned char t[10]={0,1,2,3,4,5,6,7,8,9};
unsigned short *w;
w = (unsigned int*) &t;
printf ("%d ", *w++);
printf ("%d ", *w++);
printf ("%d ", *w++);
```

Wynik:

256 770 1284

15.Uzupełnij kod funkcji zwracającej max z dwóch liczb

```
int max(int *a, int* b){
        .
        .
        .
{
```

```
int zm;
zm = (*a > *b) ? *a : *b;
return zm;
```

16. Zmienna wskaźnikowa t wskazuje na tablicę 10 liczb typu integer. Zwiększ wartość każdego elementu tablicy o 1.

```
int tab[10]={0,1,2,3,4,5,6,7,8,9};
int* t=(int* )&tab;
int i;
for (i=0; i<10; i++){
(*t++)++;
printf("%d ", tab[i]);
}
```

17. Co będzie wydrukowane na ekranie?

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

union test
{
int x1;
unsigned chr t[4];
};

int main(){
unin test ut;
ut.x1=128;
printf("  %02d  %02d  %02d  %02d" , ut.t[0], ut.t[1], ut.t[2], ut.t[3]);
getche();
}
```

Wynik:

```
128  00  00  00
```

18. Napisz część programu otwierającą do zapisu plik a.txt i sprawdzająca czy plik został otwarty prawidłowo.

```
FILE f;

f = fopen("test35.txt", "r");
if ( f == NULL )
{
```

```
printf("\n Nie mogę otworzyć pliku");
exit(1);
}
else{
//operacje na pliku
fclose(f);
}
```

19. Co pojawi się na ekranie?

```
#include <iostream.h>
#include <conio.h>

void suma(int x, int y, int *s),
int main()
{
int a=10;  int b=20;  int c=0;
suma(a,b,&c);
cout << "a= " << a << ", b= " << b << ", c= << c;
getch();
}

void suma(int x, int y, int *s)
{
*s=x+y;
}
```

Wynik:

a=10, b=20, c=30

20. Korzystając z poleceni Malloc zdefiniuj 10 elementową tablicę dynamiczną i wypełnij ją zerami.

```
int *wsk= (int*)malloc(10*sizeof(int));
if (wsk != NULL){
for (int i=0; i<10; i++) wsk[i]=0;
for (int i=0; i<10; i++)
cout << "\nelement [" << i << "] = " << wsk[i];
free(wsk);
}
```