Podstawy programowania w C – materiały dla ucznia:

Tematy lekcji:
1 Język C: edycja i kompilacja programów. Ogólna struktura i budowa programu.
2 Podstawowe elementy języka C – słowa kluczowe, instrukcje podstawowe. Przykłady prostych programów.

1) Przypomnienie wiadomości z kompilacji przy pomocy BCC32:
Edycja programu przy pomocy edytorów typu ***Notepad, Edit, Notepad ++, edytora Dev C++.***
Pliki źródłowe języka C mają rozszerzenie **C** a języka C++ - rozszerzenie **CPP**.
Kompilacja przy pomocy BCC32 – musi być ustawiona ścieżka dostępu do programu C:\BCC55\BIN\BCC32.exe :W DOS: PATH – sprawdzamy – jeśli nie ma to wpisuje się polecenie: **PATH=%PATH%;C:\BCC55\BIN**
lub lepiej na stałe: zmienne środowiskowe, PATH, edytuj – dopisać w zmienne PATH na końcu C:\BCC55\BIN i OK .
Kompilacja programu z **Borland C++ Compiler**: **BCC32 program.c** (w C) lub **BCC32 program.cpp** (wersja w C++).
Program wykonywalny po skompilowaniu w środowisku DOS, Windows ma rozszerzenie EXE.

2) Środowisko Turbo C:
Wersje: Turbo C++ version 1.01 w katalogu C:\Turbo – program C:\Turbo\Bin\TC.exe)
Options: Environment (otoczenie) Directories – Output Directory: C:\Uczen\Nazwisko lub C:\UCZEN\NI (pierwsze litery nazwiska, imienia, bez polskich liter), np. C:\Uczen\BL – Balzam Łukasz).
Turbo C++ version 3.00 – dodatkowo opcja Source Directory – Katalog źródłowy.
Właściwości skrótu programu: Ekran – Pełny ekran lub Okno.
Uruchomienie: Naciśnięcie 2x myszką na skrót programu lub przejście do katalogu programu i uruchomienie TC.EXE.
Menu programu: **File** – **Plik**: **Edit – edycja, Search**– **szukaj, Run – uruchom, Compile**: **kompiluj**, **Debug -** debugowanie **Project –** projekt, **Options – opcje, Window –** okno, **Help – pomoc** → Help, Index, wpisujemy szukane słowo, np. void i naciskamy Enter
Operacje najważniejsze w TC: Kompilujemy program ALT C, Build lub **Alt F9,**
Uruchamiamy program ALT R lub **CTRL F9,** By zobaczyć wyniki, **Alt F5**.

3) Ogólna struktura programu w języku C:
/* **Nagłówek programu** - komentarz: (nazwa, autor, data, kompilator, uwagi itp.) */
#include (**preprocesor** - włączenia tekstowe), np. *#include <stdio.h>*
#define (**preprocesor** - stałe makroinstrukcje), np. *#define PI 3.14*
Stałe, zmienne globalne (przed main), np. *const float PI=3,14; float liczba1; int*
Prototypy funkcji - deklaracje, np. *int dodawanie(float a, float b);*
Funkcja main() *{ treść_funkcji }*
Funkcje (definicje funkcji), np. *int dodawanie(float a, float b) {return a+b; }*
Preprocesor przetwarza wiersze programu rozpoczynające się znakiem **#**.
Taki wiersz nazywamy **dyrektywą** preprocesora. Podstawową **dyrektywą** jest #include, umożliwiająca dołączenie do programu pliku o podanej nazwie. Ma 2 postacie**: #include <nazwa>** i **#include „nazwa”**

4) Przykładowe programy na powitanie w 2 wersjach: C i C++ :
Program w języku **C**

```
/* Komentarz: plik hello.c */
#include <stdio.h> /* komentarz - dołącz. pliku nagłówkowego stdio.h potrzebnego do użycia funkcji printf */
int main (void) /* funkcja główna*/
{ /* początek */
printf ("Hello World!"); /* funkcja printf() wyświetla napis */
return 0; /* funkcja main() zwraca 0 */
} /* koniec funkcji głównej */
```

Po skompilowaniu uruchomić obejrzeć wyniki Alt F5.
Następnie uruchomić w linii DOS: **hello.exe > hello.txt** – skierowanie wyników do pliku hello.txt. Obejrzeć wyniki w pliku przy pomocy polecenia **TYPE** hello.txt lub otworzyć w **Notepad** (notatnik).
Program w języku **C++**

```
//Program hello.cpp - komentarz jednowierszowy w C++
#include <iostream.h> // preprocessor - dołączenie biblioteki systemowej do funkcji cout – input #include <conio.h> //
dołączenie pliku nagłówkowego <conio.h> do funkcji getch()
int main()          // funkcja główna
{
cout << "Hello World!"; // wyświetlenie napisu
getch();            //program czeka na dowolny klawisz
return 0;           // funkcja główna zwraca 0
}
```

Podstawowe elementy języka C

- **zestaw znaków; nazwy i słowa zastrzeżone** (np. **auto break case char const goto if int long register unsigned void volatile while**)
- **typy danych** (np. int, char, float, double); **stałe** (np. 121, 23L, 021, 0x12, 25L, 'a', ***const int a=10;***)
- **zmienne i tablice** (np. *float zm1, zm2; double tab[10];* **) ; deklaracje** (np. *int i, j; double k; char znak; int main()*)
- **operatory:** m.in. **matematyczne: +, -, *, /, %; przypisania =, inkrementacji ++ i dekrementacji --,**
- **wyrażenia -** to co zwraca element, np. 3.2, a=a+b; k==m; y=x=a+b; b=a[j+1];
- **instrukcje -** fragmenty tekstu programu powodujące jakąś czynność komputera podczas wykonywania programu,

Instrukcje podstawowe:
#include – dołączenie pliku nagłówkowego, podstawowy **#include <studio.h>** w C i **#include <iostream.h>** w C++ - do wydruku i wprowadzenia danych; **main()** – funkcja główna – musi być, **return** – zwracanie wartości wyniku, **getch()** – oczekiwanie na naciśniecie klawisza, **const** – deklaracja stałych, np. const int PI=3.14; **int** – deklaracja zmiennych całkowitych, **float** – rzeczywistych, np. int a; Wyjście programu (wydruk na ekranie): **printf()** – wydruk formatowany, **puts()** – wydruk tekstu, **puchar()** – wyprowadzenie znaku, ***cout*** << *wydruk w C++*.
Wprowadzanie danych: scanf() – wczytanie danych – formatowane, **gets()** – wczytanie łańcucha znaków, **getchar()** – wprowadzenie znaku, ***cin*** >> *wprowadzenie danych w C++;* **clrscr()** – kasowanie ekranu (w #include <conio.h>,

Przykłady programów

1 Obliczenie pola prostokąta

1a. Wersja prosta – dane w programie

```
/*  program pprost1.c  */
#include <stdio.h>                    /* dołączenie plików nagłówkowych */
#include <conio.h>            /* plik do funkcji getch() */
int main()
{
float a, b;             /* deklarujemy zmienne przechowujące boki prostokąta */
  float pole;           /* deklarujemy zmienną zawierającą wynik obliczeń */
  a = 5; b =10;         /*  przypisujemy zmiennym  wartości */
  pole = a * b;         /* wyrażenie - obliczamy pole prostokąta  - tu równe 50 */
  printf("Pole = %f\n",pole);           /* wydruk pola i przejście do nowej linii \n */
  getch();                      /* program czeka na naciśniecie klawisza*/
  return 0;
}
```

1b. Obliczenie pola prostokąta w wersji z wprowadzaniem danych i funkcją

```
/* ppros4.c – Obliczenie pola prostokąta - z funkcją i wprowadzaniem danych*/
#include <stdio.h>  /* dołączenie pliku nagłówkowego stdio.h potrzebnego do użycia funkcji printf  */
#include <conio.h> /* dołączenie pliku nagłówkowego conio.h do użycia funcki getch() */
float PoleProstokata(float bok1, float bok2);     /* zapowiedz funkcji – 2 argumenty, zwraca pole typu float */
int main()            /* funkcja glowna */
{
  float a, b, p;
  puts("Obliczenie pola prostokata o bokach a i b ");
  printf("Wprowadz a   ");    scanf("%f", &a);      /* wprowadzenie a */
  printf("Wprowadz b   ");   scanf("%f", &b);      /* wprowadzenie b */
  p= PoleProstokata(a, b);                          /* wywołanie funkcji z parametrami a i b */
  printf("Pole prostokat o bokach %f i %f = %10.2f \n", a, b, p);
  printf("\nNacisnij cos ");    getch();                /* czeka na znak */
  return 0;
}
float PoleProstokata(float bok1, float bok2)      /* Funkcja - definicja */
{   /* w tym miejscu bok1 jest równy a,    natomiast bok2 jest równy b */
  float wynik;
  wynik = bok1 * bok2;
  return wynik;
}
```

2 Dane osobowe

```
/ * Program daneos1.c */
#include <stdio.h> /* plik nagłówkowy */
#include <conio.h>
int main(void)  /* funkcja glowna, brak argumentow, zwraca typ calkowity int */
{
            char imie[20];        /* deklaracja tablicy znakow – łańcuch – tekst na 19 znaków */
            int i;                /* deklaracja zmiennej całkowitej i */
            printf("\nPodaj swoje imie ");  /* wydruk napisu na ekran,\n – nowa linia */
```

```
        gets(imie);                 /* wprowadzenie imienia */
        puts("Ile masz lat? ");     /*  wydruk napisu – funkcja puts() */
        scanf("%d",&i);             /* wprowadzenie lat */
        printf("\n%s ma %d lat.",imie, i);      /* wyświetlenie imienia i lat */
        getch();            /* czeka na klawisz */
        return 0;           /* funkcja główna zwraca 0 – pomyślny koniec */
}
```

3 Kalkulator

```
/* Program kalk1.c - kalkulator */
#include <stdio.h>
#include <conio.h>
int main()                  /* funkcja główna */
{
int a,b;                    /* deklaracja zmiennych całkowitych a i b */
int suma,roznica,iloczyn;
float iloraz;               /* deklaracja zmiennej rzeczywistej */
clrscr();                   /* kasowanie ekranu */
printf("Prosty kalkulator\n");
printf("\nPodaj liczbe a: ");        scanf("%d",&a);          /* wczytanie liczby a */
printf("Podaj liczbe b: ");         scanf("%d",&b);          /* wczytanie liczby b */
suma=a+b;  roznica=a-b; iloczyn=a*b;
iloraz=(float)a/(float)b;           /* operator rzutowania w dzieleniu */
printf("\nWyniki dzialan:\n");      /* Wydruk wynikow */
printf("\nSuma:   %d  ",suma); printf("\nRoznica: %d  ",roznica);
printf("\nIloczyn: %d  ",iloczyn); printf("\nIloraz:  %f  ",iloraz);
getch();                    /* czeka na naciśniecie klawisza */
return 0 ;
}
```

4 Obliczenie pola koła: Algorytm: P=Pi*r*r

Programy źródłowe w C – rozszerzenie C

1) Wersja najprostsza, program źródłowy **polkola1c**, w jednej linii, mało czytelny, bez komentarzy.

```
include <stdio.h>  main()  {  float PI=3.141593;   printf("%f",PI*5*5);   }
```

2) Wersja zmieniona – stała, zmienna, czyszczenie ekranu

```
/* program polkola2.c */
/* dyrektywy załączające tzw. nazwane pliki */
#include <stdio.h>
#include <conio.h>
/* funkcja główna */
main()
{
  const float PI=3.141593;   /* stala */
  float r=5.0;          /* zmienna */
  clrscr();             /* kasowanie ekranu - określone w conio.h */
  printf("Pole kola o promieniu %.0f = %7.3f\n",r, PI*r*r);       /* wydruk, \n – nowa linia */
  getch();              /* czeka na nacisniecie klawisza - w stdio.h */
}
```

3) Wersja z wprowadzeniem promienia i formatowaniem wydruku

```
/* program polkola3.c */
/* dyrektywy załączajace tzw. nazwane pliki */
#include <stdio.h>
#include <conio.h>
/* funkcja glowna */
main()
{
  const float PI=3.141593;   /* stala */
  float r, p;           /* zmienne – deklaracja */
  clrscr();             /* kasowanie ekranu - określone w conio.h */
  puts("Obliczenie pola kola o promieniu r ");              /* Napis na ekranie */
  printf("Podaj r => ");    scanf("%f",&r);  /* wczytanie r */
  p=PI*r*r;             /* obliczenie pola - wyrazenie */
  printf("Pole kola o promieniu %.0f = %7.3f\n",r, p);  /* wydruk */
  getch();              /* czeka na naciśniecie klawisza - w stdio.h */
}
```

4) **Wersja program w języku C++** (funkcje **cout** << wydruk wyników
oraz **cin** >> - wprowadzenie danych, funkcja własna pole_kola()

```
//  Polkola4.cpp – komentarz jednoliniowy
 #include <iostream.h> // inny plik nagłówkowy do funkcji cout i cin w C++
#include <conio.h>
  float pole_kola(float r)      // funkcja zdefiniowana w programie przed funkcją main()
  {   float p;    p = 3.1415 * r * r;     return p;   }
  void main(void)  // funkcja główna
  {
   float a, p;
   cout << "Podaj promien: ";    cin >> a;
   p = pole_kola(a);     cout << "Pole kola wynosi:  " << p << endl;  getch();
  }
```

Szablon programu do obliczeń geodezyjnych - szablgeo.c

```
/* Program szablgeo.c */
/* preprocesor: #include - wlaczenia tekstowe bibliotek */
#include <stdio.h> /* prawie zawsze c C, np. do printf() */
#include <conio.h> /* np. do getch() */
#include <math.h>
/* #define - stale makroinstrukcje */
#define PROGRAM "program.c"
#define NAZWISKO "Nowak Jan"
#define ROK 2011
#define SZKOLA "Sztygarka"
#define KLASA "2BG"
#define NL printf("\n");
#define TAB putchar('\t');
#define PI  3.141592653
#define ROG  63.66197724
#define ROS  57.2957795130;
#define KRESKA  puts("----------------------------------------------------------------------------------------------------")
#define KRESKA2 puts("================================================================");
#define KONIEC puts("\n\nNacisnij cos ")
#define KATRAD(g)  ((g)/(ROG))
#define KATGRAD(r) ((r)/(ROS))
/* zmienne globalne - stale, zmienne */
const double pi=3.14159265;
const double rograd=63.66197724, rostop=57.2957795130823;   /* Ro[grad]=200/PI, Ro[stopn]=180/PO */
const char szkola[]="Sztygarka";

float suma1(float l1, float l2); /* Przykładowa funkcja - deklaracja */
/* ========== Funkcja glowna * ======== */
int main()
 {
   clrscr();        printf("Program: %s \n",PROGRAM);     KRESKA2;
   puts(NAZWISKO);      puts(KLASA);     puts(SZKOLA);     KRESKA;
   printf("63.66198[grad]=%f[rad]",KATRAD(rograd));  /* przykład */
   /*     Kod programu       */
   KONIEC;     getch();       return 0;
 }
/* ========== Funkcje - definicje ======= */
float suma1(float l1, float l2)     /* Definicja funkcji */
{  return(l1+l2); }
```

Przykłady programów z różnymi instrukcjami

```
/*  Naglowek - komentarz wieloliniowy w C i C++ , //  jednolinowy w C++ */
/* Program strukt1.c – rożne dane i obliczenia   Autor:  Nazwisko Imię. Data: 3.11.2011 Kompilator Turbo C 1.01 */
/* preprocesor: #include - włączenia tekstowe bibliotek
i #define – stałe, makroinstrukcje */
#include <stdio.h> /* prawie zawsze c C, np. do printf() */
#include <conio.h> /* np. do getch() */
#include <math.h>
#define NL printf("\n");
#define PI 3.14159
#define R 10.0
#define KRESKA puts("-----------------------------------------------")

/* zmienne globalne - stale, zmienne */
const float KURS_USD= 3.40,  KURS_EURO= 4.50;

float suma_float(float l1, float l2);           /* zapowiedź funkcji */

float pole_kola(double r)          /* definicja funkcji */
{
 double pole, pole1;
 pole=PI*r*r;
 puts("\nFunkcja pole_kola()");
 printf("\Pole z PI   =%f",pole);
 pole1=M_PI*r*r;
 printf("\nPole z M_PI =%f ",pole1);  NL; NL;
 return pole1;
}

int main()                /* funkcja główna */
  {
    int a, b=3, suma, reszta;
    float kw_zl, kw_usd;
    clrscr();
    puts("Struktura programu w C - przyklad\n");
    KRESKA;
    puts("Naglowek");
    puts("Preprocesor - #include, #define");
    puts("Zmienne globalne i prototypy funkcji lub definicje funkcji");
    puts("Funkcja main()");
    puts("Funkcje - definicje");     NL;     KRESKA;
    puts("Przyklady wynikow");     NL;
    puts("Funkcja glowna");     NL;
    a=10;      kw_zl=10;      kw_usd=kw_zl/KURS_USD;
   /* suma=a+b;  // bez funkcji  */
    suma=  suma_float(a,b);
    reszta=a%b;
    printf("Suma %d +  %d =  %d \n",a,b,suma);
    printf("Suma 2.50 + 3.40 = %f",(2.50+3.40));     NL;
    printf("Reszta z dzielenia %d  przez  %d =  %d ",a,b,reszta);     NL; NL;
    printf("Kwota %.2f PLN = %.2f USD ",kw_zl, kw_usd);     NL;
    printf("Pole kola o promieniu %f = %f ",R, pole_kola(R));     NL;
    getch();
    return 0;
  } /* koniec funkcji glownej */

/* ----  definicja funkcji  ----*/
float suma_float(float l1, float l2)
{
 return(l1+l2);
}


/* Program Daneucz.c    - dane ucznia*/
/* preprocesor: #include - wlaczenia tekstowe bibliotek
i #define - stale makroinstrukcje */
#include <stdio.h> /* prawie zawsze c C, np. do printf() */
#include <conio.h> /* np. do getch() */
#include <math.h>
#define PROGRAM "program.c"
#define NAZWISKO "Nowak Jan"
```

```
#define ROK 2011
#define KLASA "2BG"
#define NL printf("\n");
#define TAB putchar('\t');
#define PI 3.14159
#define KRESKA puts("-------------------------------------------------")
#define KRESKA2 puts("=================================================");
#define KONIEC puts("\nNacisnij cos ")

/* zmienne globalne - stale, zmienne */
const int  klasa = 2;

int main()
 {
   char c1;
   char  przedmiot[20], szkola[]="Sztygarka";
   int rok_urodz;
   float ocena;
   clrscr();
   printf("Program: %s \n",PROGRAM);      KRESKA2;
   puts(NAZWISKO);      puts(KLASA);      puts(szkola);      KRESKA;
   printf("\nPodaj swoj ulubiony przedmiot (jeden wyraz):  ");      gets(przedmiot);   /* wprowadzenie imienia */
   printf("Wprowadz swoj inicjal:  ");     c1=getchar();  // wprowadzenie znaku
   printf("Podaj rok urodzenia:  ");      scanf("%d",&rok_urodz);   /* wprowadzenie roku */
   printf("Twoja srednia ocena ");      scanf("%f",&ocena);    /* wprowadzenie roku */
   KRESKA;     printf("Twoj inicjal:  ");      TAB;      putchar(c1);      NL;
   printf("\n %s  ma  %d lat.",NAZWISKO, (ROK-rok_urodz));   /* wyświetlenie imienia i lat */
   NL;
   printf("Srednia ocena z przedmiotu %s =  %.1f  ",przedmiot, ocena);
   NL;     KRESKA;      KONIEC;
   getch();
   return 0;
 }

float suma_float(float l1, float l2)
{  return(l1+l2); }
```

Szablon programu ucznia

```
/* Program Szablucz.c */
/* #include - włączenia tekstowe bibliotek */
#include <stdio.h> /* prawie zawsze  np. do printf() */
#include <conio.h> /* np. do getch() */
#include <math.h>
/* #define - stale makroinstrukcje */
#define PROGRAM "program.c"
#define NAZWISKO "Nowak Jan"
#define ROK 2011
#define SZKOLA "Sztygarka"
#define KLASA "2BG"
#define NL printf("\n");
#define TAB putchar('\t');
#define PI 3.141592653
#define KRESKA puts("------------------------------")
#define KRESKA2 puts("================");
#define KONIEC puts("\n\nNacisnij cos ")
#define KATRAD(g)  ((g)/(ROG))
/* zmienne globalne - stale, zmienne */
const double pi=3.14159265;
const char szkola[]="Sztygarka";
float suma1(float l1, float l2);  /* Przykladowa funkcja - deklaracja */
/* ========== Funkcja główna * ======== */
int main()
 {
   clrscr();         printf("Program: %s \n",PROGRAM);     KRESKA2;          puts(NAZWISKO);
```

```
  puts(KLASA);    puts(SZKOLA);    KRESKA;    /* -- dalszy kod programu glownego --  */
  KONIEC;
  getch();    return 0;
 }
float suma1(float l1, float l2)      /* Definicja funkcji */
{  return(l1+l2); }
```

Podsumowanie:

- Programy w C mają rozszerzenia plików C a w C++ rozszerzenie CPP.
- Do pisania kodu źródłowego używamy edytora tekstowego by plik wynikowy zawierał tylko znaki ASCII (bez znaków formatowania jak np. styl tekstu w Wordzie).
- Polecenia kończymy średnikiem. Kod funkcji umieszcza się w nawiasach klamrowych.
- W jednej linii mogą 2 instrukcje ale nie jest to zalecane.
- Komentarze typu /* … */ w C i C++ mogą być wielo-liniowe oraz typu // w jednej linii tylko w C++ (choć są zwykle tolerowane obecnie też w C).
- Program zawiera zawsze: Funkcję **main**() zwykle w wersji: int main() { … return 0} - zwraca 0 gdy wykonana poprawnie. Zwykle program zawiera nagłówek, dyrektywy preprocesora , zwłaszcza #include, czasem #define – stałe i makroinstrukcje. Przed funkcją główną mogą być zadeklarowane lub zdefiniowane zmienne globalne oraz funkcje.
- By obejrzeć wyniki w Turbo C (jeśli nie wprowadzono zatrzymania ekranu przy pomocy np. getch()), to należy przełączyć się na ekran użytkownika **Alt F5.**
- Kompilacja programów konsolowych (wykonywanych w oknie DOS) – przy pomocy BCC32:
- BCC32 program_zrodlowy lub **Alt F9** w Turbo C. Program wynikowy ma rozszerzenie EXE.
- Są w **C** **2 rodzaje komentarzy** /* wielowierszowy w **C i C++** */ i jednowierszowy w **C++** **//** - i można w nich pisać uwagi, pomijane przez kompilator.
- Aby zatrzymać pracę programu, używamy funkcji ***getch();***.
- Do czyszczenia ekranu służy funkcja ***clrscr();***.
- Na końcu funkcji **main** jest zwykle ***return 0;***.
- W tekście możemy używać tzw. znaków specjalnych, np. przejście do następnej linii ***\n.***
- **Program** składa się z ciągu rozdzielonych średnikami instrukcji położonych pomiędzy słowami kluczowymi **{** i **}**
- **Instrukcje** mogą zawierać wyrażenia oraz wywołania funkcji.
- **Wyrażenia** składają się ze stałych, operatorów i identyfikatorów.
- **Identyfikatory** są nazwami obiektów składających się na program. Mogą one zawierać litery, cyfry i znaki podkreślenia, nie mogą jednak zaczynać się od cyfr.
- Instrukcje podstawowe w C: #include, main(), return, wyprowadzanie wyników: printf(), putchar(, puts(), cout <<; wprowadzanie danych : scanf(), getchar(), gets(), cin >>

Pytania kontrolne:

Programy języka C i C++ to kompilatory czy edytory?

Czym się różni język C od C++ i jakie są rozszerzenia plików źródłowych w tych językach.

Program źródłowy a program skompilowany – rozszerzenia plików.

Środowiska programistyczne C i C++.
Jakie mogą być używane edytory do pisania programów w C – czy może to być np. Word?
Jaka funkcja musi wystąpić zawsze w języku C?

Wersje funkcji main(). Co zwraca zwykle funkcja main().

Czym kończymy instrukcje w C? Komentarze w C i C++.

Z czego składa się program? Czy nazwy identyfikatorów (stałych, zmiennych) mogą zawierać spacje?

Do czego służą dyrektywy preprocesora, jakie są najczęściej stosowane w #include..

Co to są słowa zastrzeżone w programie?

Wprowadzanie danych do programu i wyprowadzanie wyników – najważniejsze funkcje.
Do czego używamy znaku specjalnego \n?. Jak zatrzymać pracę programu? Czyszczenie ekranu w C.